VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Bakalářská práce - Domácí katalog

Bachleor thesis - Home catalog

2010 Jakub Šenk

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2010 .....

Rád bych na tomto místě poděkoval Ing. Petru Olivkovi za pomoc při psaní této bakalářské práce.

Abstrakt

Práce pojednává o vývoji softwaru, jehož úkolem je spravovat domácí sbírku filmových médií (CD/DVD). Vlastní software je přiložen na CD k bakalářské práci. Aplikaci jsem pojmenoval "Domácí katalog"

Klíčová slova: média, katalog filmů, diplomová práce, Java, MySQL

Abstract

The work is about development of software for cataloging home collection of digital movie medias (CD/DVD). Software is included to bachlelor work. Aplication has name "Home collection".

Keywords: medias, catalog of movies,Bachleor thesis, Java, MySQL

Seznam použitých zkratek a symbolů

DVD	–	Digital Versatile Disc
CD	–	Compact Disc
JIT	–	Just In Time
IDE	–	Integrated Development Environment
API	–	Application Programming Interface
AWT	–	Abstract Window Toolkit
JSP	–	Java Server Pages

Obsah

1 Úvod **5**
1.1 Členění bakalářské práce 6

2 Analýza dostupných programů **7**
2.1 Ant movie catalog 7
2.2 Seznam DVD 2008 portable 9
2.3 GCstar 11
2.4 Funkce a úloha mého programu 13

3 Výběr vhodného jazyka, frameworku a databáze **15**
3.1 Výběr vhodného jazyka 15
3.2 Výběr vhodného databázového systému 15
3.3 Výběr frameworku 17

4 Analýza a návrh vlastního řešení. Přístup ke zdrojům dat. **18**
4.1 Zadání 18
4.2 Datová analýza 20
4.3 Funčkní analýza 26
4.4 Časová analýza 32

5 Návrh grafického rozhraní. **33**
5.1 Vzhled aplikace 34

6 Realizace a programování. **36**
6.1 Fáze realizace 36
6.2 Vývojové prostředky 36
6.3 Kód aplikace 36
6.4 Vybrané partie zdrojového kódu 37

7 Uživatelská dokumentace - volba formátu. Popis instalace. **51**
7.1 Instalace programu 51
7.2 První spuštění aplikace 52
7.3 Průvodce aplikací 54

8 Testování, spolehlivost, možnosti rozšiřování. **63**
8.1 Testování a spolehlivost 63
8.2 Možnosti rozšíření 65

9 Závěr **67**

10 Reference **68**

Seznam tabulek

1 Film 22
2 Vypujcky_film 23
3 Uzivatel 23
4 Znamy 24
5 Email 24
6 Zakladni_obrazky 24
7 Nastaveni 25
8 Typ_databaze 25
9 Prihlaseny_uzivatel 25

Seznam obrázků

1 Hlavní okono programu Ant movie catalog 8
2 Hlavní okno programu Seznam DVD 10
3 Hlavní okno programu GCstar 12
4 ERD 20
5 Kontextový diagram aplikace 26
6 DFD 0.úrovně 26
7 DFD 1. úrovně 27
8 DFD 2. úrovně 28
9 DFD 3. úrovně 28
10 DFD 4. úrovně 29
11 DFD 5. úrovně 29
12 DFD 6. úrovně 30
13 DFD 7. úrovně 30
14 STD entity film 32
15 STD entity film 32
16 Přihlášení do programu 34
17 Hlavní okno aplikace 35
18 Přihlašovací obrazovka aplikace 53
19 Konfigurace připojení k MySQL databázi 53
20 Registrace nového uživatele 54
21 Hlavní formulář aplikace 55
22 Hlavní formulář aplikace 56
23 Správa vypůjček 58
24 Správa uzivatelu 59
25 Správa známých 60
26 Správa emailů 61
27 Upozornění na včas nevrácené filmy 62

Seznam výpisů zdrojového kódu

1 Konstruktor třídy Film 38
2 Načtení filmu do formuláře 39
3 Načítáme film 40
4 Načtení obsahu komponent pro uložení změněného filmu 41
5 Uložení změněného filmu 42
6 Smazání daného filmu 43
7 Vytvoření vypůjčky 44
8 Vytvoření vypůjčky 2 45
9 Prodloužení vypůjčky 46
10 Prodloužení vypůjčky 47
11 Načítání z CSFD - nalezení stránky filmu 48
12 Načítání z CSFD - nalezení informací o filmu 50

1 Úvod

Dnes, na začátku 21. století, v době, kdy lze zakoupit film na DVD i v trafice v řádu deseti a sto korun, má již prakticky každý nějakou tu sbírku filmů na těchto médiích. Ani já nejsem výjimkou, a jelikož mne znepokojoval nepořádek v mé sbírce filmů, rozhodl jsem se, že jej vyřeším pomocí počítače a softwaru k tomu určenému.

Protože nepoužívám operační systém od Microsoftu, ale používám systém alternativní (Linux a jeho distribuci Mandriva), tak je problém v tom, že většina existujících programů pro správu domácí sbírky filmů je určena pouze pro Windows a v jiných operačních systémech nefunguje . Mým cílem tedy bylo vytvořit aplikaci, která by byla spustitelná na všech dnes hojně využívaných operačních systémech (Windows, GNU/-Linux, MacOS, OpenSolaris, Solaris). Dalším nedostatkem drtivé většiny konkurenčních programů je, že nepočítají s víceuživatelským přístupem, což je minimálně nepraktické (nelze adekvátně rozlišit mezi vlastníky jednotlivých DVD, což se často hodí).

Také jsem se rozhodl, že aplikace bude postavena na otevřených technologiích a já mohl na aplikaci pokračovat i později, po ukončení studia. Kdyby aplikace nebyla postavena na otevřených technologicích, musel bych si potřebné vývojové nástroje a další potřebný software nakoupit, což by se velice prodražilo. V dalších verzích aplikace se počítá i s webovým rozhraním a s dalšími funkcemi, o čemž se zmiňuji v kapitole 8.2.

Z výše uvedených důvodů jsem se tedy rozhodl naprogramovat k tomuto účelu program vlastní, který bude výše uvedené problémy řešit, bude tedy multiplatformní (viz definice 1.1), přenositelný (viz definice 1.2) a víceuživatelský (viz definice 1.3). Předchozí větou jsem prakticky definoval jazyk, kterým bude aplikace psána, protože výše uvedené zajistí prakticky pouze jazyk Java, který jako jediný umožní, aby byla aplikace multiplatformní a zároveň přenositelná, u konkurečních jazyků by musela pro každou platformu existovat speciální verze programu. Popis funkcí programu pak naleznete na straně 13.

Ještě před zahájením vývoje aplikace jsem provedl analýzu konkurenčních programů, kterou si můžete přečíst v kapitole „Analýza dostupných programů“, viz. kap. 2.

Definice 1.1 *Multiplatformní software je takový, který může fungovat na více než jedné platformě, např. tedy na MS Windows a GNU/Linuxu. [3].*

Definice 1.2 *Přenositelný software je takový, který se dá přenášet mezi počítači bez nutnosti instalace či kompilace.*

Definice 1.3 *Víceuživatelská aplikace je taková aplikace, která umožňuje, aby aplikaci používalo více lidí, a každý z nich měl vlastní profil, vlastní záznamy a vlastní nastavení.*

Vlastní text bakalářské práce je psán v sázecím systému LaTeXpomocí textového editoru Kile nainstalovaném v Mandrivě 2010.

1.1 Členění bakalářské práce

Práce je je dle zadání rozčleněna do 7 kapitol, do členění není zahrnut Úvod a Závěr práce:

1. Analýza dostupných programů, viz str. 7
2. Výběr vhodného jazyka, frameworku a databáze, viz str.15
3. Analýza a návrh vlastního řešení. Přístup ke zdrojům dat, viz str. 18
4. Návrh grafického rozhraní, viz str. 33
5. Realizace a programování, viz str. 36
6. Uživatelská dokumentace - volba formátu. Popis instalace, str. 51
7. Testování, spolehlivost, možnosti rozšiřování, viz str. 63

V části „Analýza dostupných programů“ (str. 7) se zabývám průzkumem mezi konkurenčními programy, které jsou na trhu. Samozřejmě jsem vybral jen několik z nich. U každé zmiňované aplikace se zabývám jejími vlastnostmi, zápory a co se mi naopak líbilo a čím jsem se tedy inspiroval. Také zde uvádím výčet hlavních funkcí programu.

V části „Výběr vhodného jazyka, frameworku a databáze“ (str. 15) zdůvodňuji, proč jsem si vybral Javu jako programovací jazyk, MySQL jako databázový server a také zde uvádím seznam použitých externích knihoven, které nejsou součástí Java API.

V další části, nazvané „Analýza a návrh vlastního řešení. Přístup ke zdrojům dat.“ (str. 18) je provedena klasická analýza, jakou jsme se učili v předmětech TZD a DAIS. Je rozdělena na tři části - „Analýza datová“ (str. 22), „Analýza funkční“ (str. 26) a Analýza časová (str. 32).

Následující část, část Návrh grafického prostředí (str. 33) obsahuje informace o tvorbě vzhledu programu a použitých technologích k jeho tvorbě.

Jedna z nejdelších kapitol je kapitola „Realizace a programování“ (str. 36) obsahuje nepřeberné množství informací o vlastní tvorbě programu. Nejprve informuje čtenáře o členění zdrojových kódu do balíčků a čtenáře informuje i o pozadí tvorby programu. Taktéž obsahuje vybrané partie zdrojového kódu včetně popisu (str. 37).

V následující kapitole pojmenované „Uživatelská dokumentace - volba formátu. Popis instalace.“ (str. 51) se čtenář seznamuje s prostředím programu. V této části práce je nejprve popsána instalace programu, a poté i většina podstatných funkcí programu.

V poslední kapitole práce, kterou jsem pojmenoval “Testování, spolehlivost, možnosti rozšiřování.“ (str. 63), se pak čtenář seznamuje s metodikou testování programu. Nedílnou součástí této kapitoly jsou také informace o případném budoucím rozšíření aplikace (str. 65).

2 Analýza dostupných programů

V této kapitole jsem nejprve provedl analýzu nejznámějších konkurenčních programů, které jsou dostupné na Internetu. Samozřejmě jsem nemohl vyzkoušet všechny programy tohoto typu, ale vybral jsem tři z nich, které mě osobně oslovily nejvíce.

Hlavním účelem analýzy tak je, abych se poučil z jejich chyb a případně se inspiroval zajímavými funkcemi. V další části této kapitoly se zmiňuji o tom, jaké bude mít můj program funkce a vlastnosti a proč.

Předmětem analýzy se tak staly tyto tři programy: Ant movie catalog, (kapitola 2.1), Seznam DVD 2008 Partable (kapitola 2.2) a GCstar (kapitola 2.3).

2.1 Ant movie catalog

Jedná se pravděpodobně o nejznámější program, je napsán v Pascalu a je dostupný pouze pro platformu Windows.

2.1.1 Vlastnosti aplikace

- otevřený zdrojový kód, česká lokalizace
- zdarma [1]
- poměrně přehledné ovládání, ale s výhradami (viz níže)
- užitečné funkce (statistiky, vyhledávání filmů na webu)

2.1.2 Zápory aplikace

- pokud vyhledávám film dle původního jména na Internetu (pomocí např. skriptu pro csfd.cz), ve výsledcích se preferují názvy české, což je někdy na škodu (když neznám např. český název, nebo je pro dva filmy stejný)
- při výběru filmu z nalezeného seznamu filmů vidím jen název a rok natočení, lepší by bylo, kdyby bylo zobrazeno více.
- složitý výběr serveru, na kterém se bude daný film hledat -> musí se při každém vyhledávání vybírat daný server -> zdržuje
- vyhledání filmu se provádí pomocí volby z menu, takto častá akce by měla být lépe dosažitelná -> třeba tlačítkem, které by bylo součástí formuláře pro přidání filmu
- nepřehledné (nepraktické) ovládání (v některých chvílích)
- pouze jeden uživatel = pouze jedna DB, leda mezi nimi přepínat (nepraktické)

[1] Program lze stáhnout z domovských stránek, http://www.antp.be/software/moviecatalog/

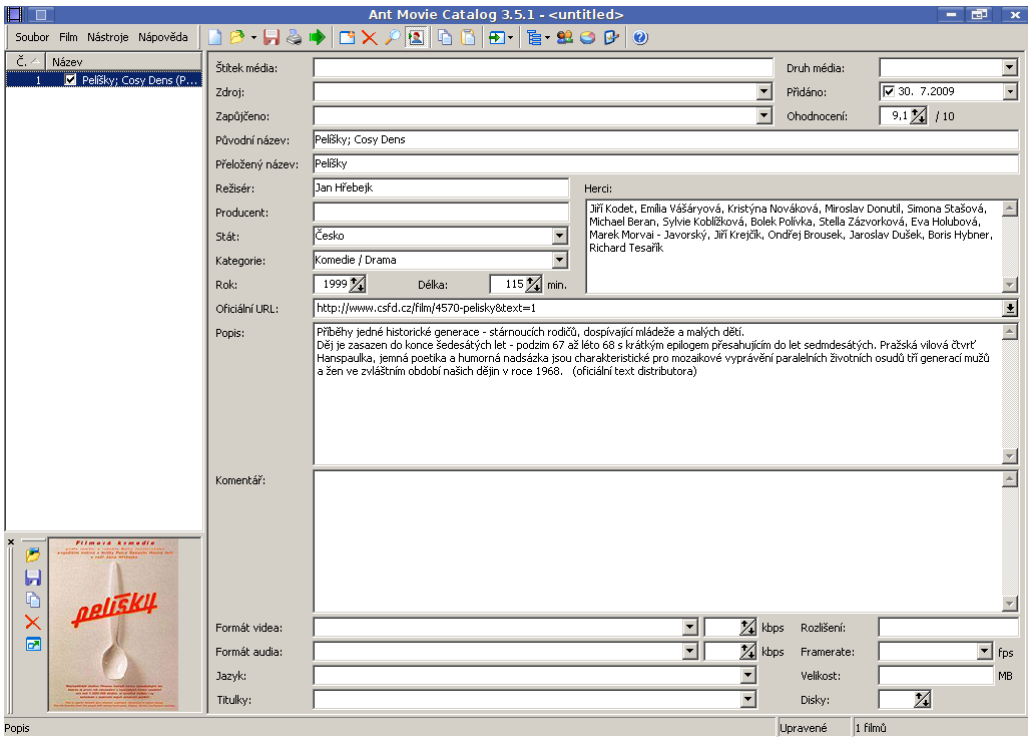
Dále pak je třeba špatně zpracována možnost půjčování média známým a podobně:

- půjčená média nejsou nijak zvýrazněna ve výčtu, vypůjčená média se dozvím pouze přes speciální seznam dostupný přes menu
- nelze určit kdy to chce vrátit, nelze posílat varovné emaily na překročení vypůjční doby

2.1.3 Čím jsem se inspiroval

- jeho vzhledem, použil jsem podobné rozvržení, např. výběr konkrétního média vlevo ze seznamu, booklet pod tímto seznamem, vpravo se zobrazují informace o médiu, a podobně. Vzhled aplikace je pochopitelně již však zásadně jiný.
- některými funkcemi jako vyhledávání informací o filmech na Internetu, také použití statistik, načítání obalů z DVD a podobně. Nejvíc jsem se tedy inspiroval právě tímto programem.

2.1.4 Screenshot



Obrázek 1: Hlavní okono programu Ant movie catalog

2.2 Seznam DVD 2008 portable

2.2.1 Vlastnosti aplikace

- shareware[2] - omezené vyhledávání na fdb.cz a počet položek v databázi
- projekt fdb.cz - dobrá integrace s jejími službami (vyhledávání ve filmech)
- česká lokalizace
- verze 6.21
- hodnocení filmů v grafické formě
- u každého titulu lze nastavit, že daný film daný uživatel neviděl
- detail filmu je v HTML podobě
- lze přidávat komentáře k filmu na fdb.cz přímo z prostředí programu
- podporuje export na FTP server, nelze s takovou databází ale pracovat, nejdříve se musí stáhnout
- podporuje mnoho různých statistik, třeba třídění dle počtu filmů s daným dabingem, titulkami, žánrem a podobně, informace o celé databázi, výdělku za půjčování a podobně

2.2.2 Zápory aplikace

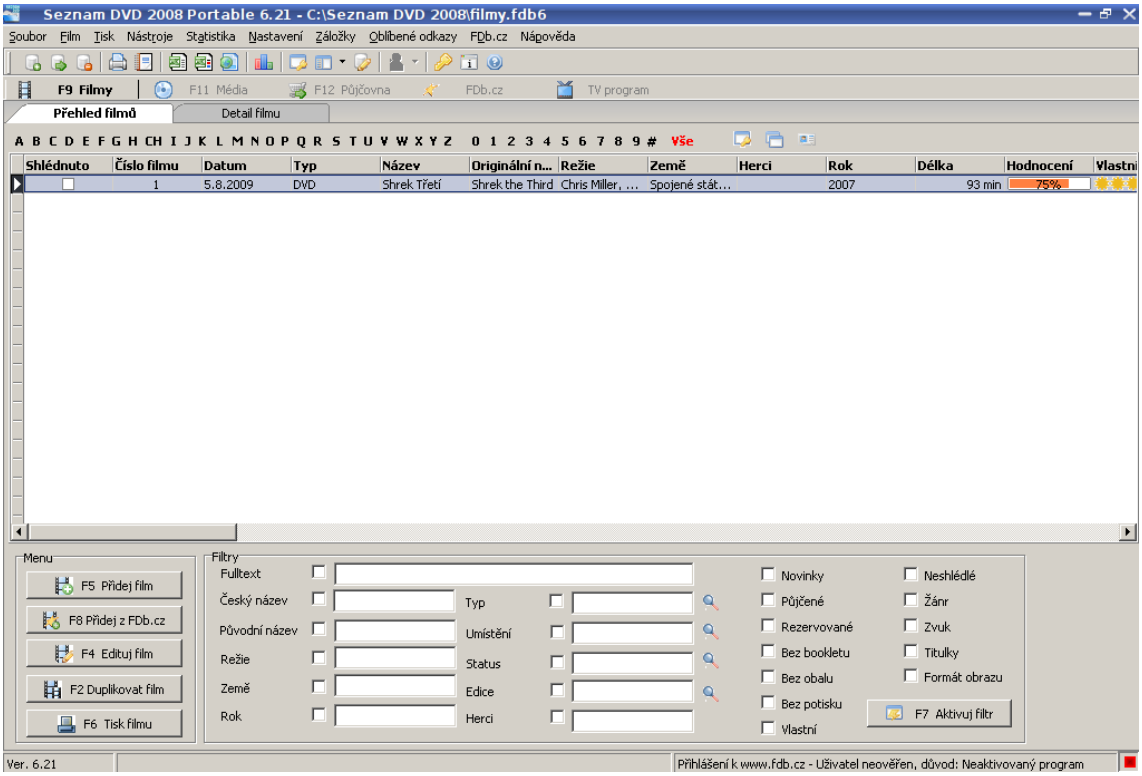
- mírně nepřehledné vlivem přemíry informací o filmu a funkcí programu
- hodně omezená shareware verze, např. nelze vyhledávat na Internetu jakýkoli film, ale jen pár přednastavených
- seznam s filmy je moc široký, jelikož obsahuje mnoho údajů, lepší je zobrazit jen nejzákladnější s tím, že další se zobrazí po kliknutí
- jedná se spíše o software pro videopůjčovny

2.2.3 Čím jsem se inspiroval

- daleko přehlednější zakládání databáze než v případě Ant movie catalog
- díky nepřehlednosti v programu, způsobené přemírou informací o filmu jsem si uvědomil, že není dobré to přehánět s množství informací o filmu

[2] Trial verzi programu lze stáhnout z domovských stránek, http://www.fdb.cz/seznamdvd/

2.2.4 Screenshot



Obrázek 2: Hlavní okno programu Seznam DVD

2.3 GCstar

2.3.1 Vlastnosti aplikace

- opensource, multiplatformní
- verze 1.5 beta
- v češtině
- podporuje i hry, software, knihy a podobně, nejen filmy
- podporuje import i z konkurenčních programů (Ant movie catalog a podobně)

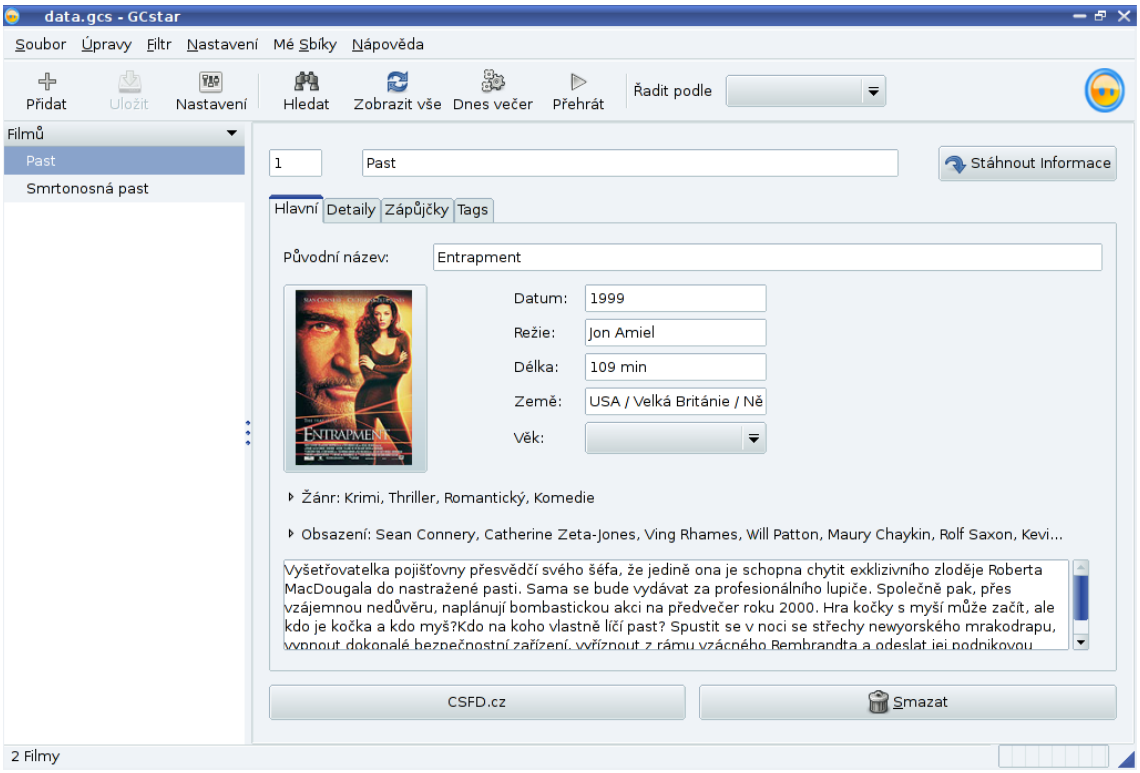
2.3.2 Zápory aplikace

- nenachází filmy v csfd.cz, ač tato funkce je dostupná a vše je nastaveno správně a film na csfd.cz je, třeba Pelíšky, ale toto se stává jen výjimečně
- nepřišel jsem na způsob, jak mít otevřeno více databází najednou
- je sice přenosný a multiplatformní, nevýhodou ale je, že pro každou platformu je zvláštní instalační balíček

2.3.3 Čím jsem se inspiroval

- přehledné, informace jsou pouhým textem (žádná editační políčka zabírající prostor), pokud je chce uživatel editovat, klikne na danou sekci a ta se mu „rozevře“ i s editačními poli - úspora místa

2.3.4 Screenshot



Obrázek 3: Hlavní okno programu GCstar

2.4 Funkce a úloha mého programu

Účelem aplikace je tedy udržovat pořádek v domácích sbírkách filmů na DVD či CD, případně i na VHS. Z tohoto tvrzení také vyplívá, že aplikace je určena pouze pro domácí použití a není vhodná pro videopůjčovny, protože obsahuje například pouze základní funkce pro půjčování médií a podobně.

Můj cíl by se dal shrnout do jedné věty asi takto: *Multiplatformní přenositelná aplikace dostupná odkudkoli pro správu domácích médií.*

Hlavním účelem mé aplikace tedy bude správa domácí sbírky různých CD a DVD tak, aby v nich měl potencionální uživatel pořádek a vždy věděl, co vlastně má, zda-li dané médium má někdo půjčeno a jak dlouho. Aplikace bude spravovat pouze filmové média.

Počet spravovaných médií samozřejmě nebude nijak omezen, jediným omezením je velikost úložného prostoru na databázovém stroji. Aplikace bude umět základní paletu funkcí, jako např. práva médií a jejich vkládání, mazání a editování, základní export a statistiky, tisk, vyhledávání, načítání z Internetu, zobrazování bookletů a jejich ukládání a podobně. O těchto základních funkcích se budu zmiňovat až v uživatelské příručce (str. 51), nyní budou představeny pouze nejzajímavější funkce plánovaného programu.

Aplikace je spustitelná na jakémkoli moderním počítači, na kterém dokáže běžet jeden z několika nejpoužívanějších operačních systémů, čili MS Windows, GNU/Linux, MacOS, *BSD a Solaris. Mým cílem pak je, aby se nemusel uživatel starat o to, jaký OS je nainstalován, jednoduše připojil flashdisk, spustil by aplikaci a přihlásil se k databázi, kterou by měl na MySQL serveru někde na Internetu, případně lokálně nainstalovanou.

Navíc by potencionální uživatel nemusel mít na flashdisku X různých verzí tohoto programu, kdy každá by byla pro jiný OS, místo toho by měl verzi jednu, která by sama rozpoznala, o jakou platformu jde.

Když jsem prováděl analýzu konkurenčních programů, nemohl jsem si nepovšimnout jednoho z velkých nedostatků všech aplikací, které jsem analyzoval. Žádná z nich nedovoluje mít databázi na serveru dostupném na Internetu, všechny používají pouze lokální databázi. Databáze uložená na serveru má tu výhodu, že data jsou odkudkoli dostupná, typickým příkladem pak je třeba situace, kdy se uživatel chce pochlubit se svou sbírkou filmů a dalších médií kamarádovi na jeho počítači. V tomto případě tudíž stačí, aby si uživatel stáhl aplikaci (pokud ji nemá u sebe) a přihlásil se k serveru, na kterém má databázi, společně s tímto se i stáhne nastavení aplikace. Lokální databáze povětšinou nedovolují multiuživatelský přístup.

Aplikace je solidním základem i pro případné komerční nasazení do praxe. Aplikace jako taková by byla zdarma. Uživatel by si pak mohl vybrat, zda-li si chce nainstalovat doma MySQL server nebo si jen stáhnout aplikaci a využívat (za nějaký rozumný poplatek) můj MySQL server, který by byl ihned k použití. Bonusem by pak byl fakt, že by tato databáze byla dostupná kdekoli, stačilo by připojení na Internet. Později by byly vytvořeno i webové rozhraní a klient pro mobilní telefony a uživatel by tak mohl přistupovat ke své sbírce filmů prakticky odkudkoli, třeba ihned poté, co si film koupí v trafice, tak jej pomocí mobilního telefonu přidá i do své sbírky. Vše by bylo napsáno pomocí jednoho

programovacího jazyka - Java. O možnostech budoucího rozšíření programu se zmiňuji v patřičné kapitole, viz. strana 65.

Samozřejmě aplikace bude umět všechny základní funkce, které umí i konkurenční produkty, čili načítat obaly, statistiky, tisk seznamů, půjčování médií kamarádům a známým, posílat upomínky o překročení doby půjčení a podobně.

3 Výběr vhodného jazyka, frameworku a databáze

3.1 Výběr vhodného jazyka

Při výběru programovacího jazyka (stejně tak při výběru dalších záležitostí) jsem se snažil co nejvíce dodržet své cíle, které jsem si stanovil v úvodu své Bakalářské práce (viz str. 5).

Rozhodoval jsem se mezi těmito programovacími jazyky:

- Java
- C++

S těmito dvěma programovacími jazyky mám největší zkušenosti a oba umí vše, co jsem potřeboval. Navíc se vyučují i na naší fakultě a jsou i velmi populární. Nakonec jsem se rozhodl pro Javu, protože C/C++ neumožnuje některé mé cíle.

Mým hlavním cílem je, aby aplikace byla opravdu přenosná a pro všechny podporované platformy existovala pouze jedna verze programu. Java se k tomuto velmi hodí, protože ve výsledku generuje jeden JAR balíček, který obsahuje celou aplikaci, výjimkou jsou externí knihovny, které nejsou součástí JAVA API, a které jsou k programu přidány jako samostatné soubory, další výjimkou jsou externí soubory, např. konfigurační a jiné. Každopádně pro všechny platformy je pouze jeden JAR balíček, který je spustitelný a spouští vlastní aplikaci (JAR balíček si lze představit zjednodušeně jako EXE soubor na platformě Windows). Uživatel tak zkopíruje několik souborů aplikace na flash disk a program je k dispozici kdykoli, ať se jedná o PC s Windows, MacOS nebo GNU/Linuxem.

Další nespornou výhodou a důvodem, proč jsem použil právě Javu je i fakt, že programátor se nemusí při programování zajímat o několik základních věcí, jako například o uvolňování paměti, o vše se totiž stará sama Java díky použití tzv. Garbage Collectoru. Tím pádem je vývoj rychlejší. Velkou roli při výběru programovacího jazyka (a i ostatních záležitostí) hrají mé osobní preference. Javu jsem si během svých studií totiž velmi oblíbil a osobně ji považuji za velmi přívětivý a dobrý programovací jazyk.

Ve prospěch jazyka C++ mluví jen fakt, že výsledný program je v drtivé většině případů rychlejší, než tentýž program v Javě. Java však během svého vývoje udělala v tomto velký pokrok a díky JIT kompilátoru [3] a dalším urychlujícím technologiím jsou rozdíly v mnoha případech zanedbatelné. Při vývoji jsem se navíc snažil o co největší efektivitu kódu.

3.2 Výběr vhodného databázového systému

Jelikož má aplikace uchovávat informace o filmech a o dalších věcech, je potřeba tyto informace někde uchovávat. Nejlepším řešením je samozřejmě použí některého z dostupných databázových systémů.

[3] Více o JIT kompilátoru se dočtete zde : http://cs.wikipedia.org/wiki/JIT

Namátkou vybrané dostupné databázové systémy:

- MySQL - zdarma, otevřená licence, existuje i placená verze enterprise
- PostgreSQL - zdarma, otevřená licence
- Oracle - placený, existuje i verze zdarma, ale ta je omezena funkčně i licenčně.
- Microsoft SQL server - placený, existuje i zdarma dostupná verze, která je ovšem stejně jako Oracle omezená
- SQLite - zdarma, nefunguje na principu klient-server, ale každá databáze je uložena do klasického souboru.

Pokud vezmeme čistě zaměření aplikace a na chvíli zapomeneme mé, v úvodu nastíněné, cíle, pak můžeme říct, že každý z těchto databázových systémů může být použit při vývoji aplikace. V následujících řádcích se pokusím, již s přihlédnutím ke svým cílům, říci, proč ten či onen databázový systém není pro mě vhodným a proč jsem jej tedy nepoužil.

Začneme tentokráte od konce. **SQLite** je jedním z používaných databázových systémů dostupných na současném trhu. Jeho výhodami je například fakt, že svůj chod nepotřebuje žádný server a nepoužívá tak principu klient-server, místo toho pro každou databázo vytvoří klasický soubor na počítači uživatele. Zprvu jsem uvažoval i o tomto databázovém systému, nakonec jsem se rozhodl jej nepoužít. Důvodů jsem měl hned několik. Jednak by aplikace nebyla moc dobře rozšiřitelná. V dalších verzích tohoto programu se totiž počítá i s webovou podobou, která by vyžadovala pravděpodobně již vyspělejší databázový systém a v tom případě by tu byl ihned problém se synchronizací dat mezi těmito dvěmi databázemi. Zásadním problém je pak fakt, že jakmile se do databáze cokoli zapisuje, tak je celá databáze uzamknuta a je tak nedostupná pro ostatní, viz literatura - [4]. Tato databáze tak je vhodná pro programy, kde se počítá pouze s jednouživatelským režimem, čili s režimem, kdy se k databázi může připojit pouze jeden uživatel.

Microsoft SQL server - jako jediný, mnou uvažovaný, databázový systém podporuje pouze systémy Windows a není tak dostupný pro alternativní systémy.

Oracle - jedná se o multiplatformní databázový systém, který je dostupný i zdarma, ač je funkčně omezen, pro potřeby aplikace by však byl dostačující. Oracle jsem si nevybral hlavně proto, že s ním nemám žádné zkušenosti, od letošního akademického roku je již Oracle v osnovách některých předmětů, já je ovšem absolovoval ještě když byl preferován MySQL. Poprvé jsem se tak setkal s Oraclem až letos v letním semestru.

PostgreSQL - otevřený objektově-relační databázový systém, který je tak trochu ve stínu MySQL.

MySQL - opět otevřený databázový systém, vyučoval se i na naší fakultě. Během výuky jsem si jej velmi oblíbil a rozhodl jsem se jej nasadit i ve své bakalářské práci. Jedná se o základní kámen technologie LAMP (Linux, Apache, MySQL, PHP), na kterém je postaveno velké množství webových serverů.

Ve výběru sehrála velkou roli moje zkušenost s danými databázovými systémy. U MySQL se mi líbí přehledné chybové hlášky, čili nalezení případné chyby je relativně jednoduché. Oracle je dle mého názoru v tomto ohledu na tom hůře, zobrazuje sice i česky varovné hlášky, ale mnohdy jsem se setkal s tím, že byly hodně zavádějící, např. chybová hláška říkala, že chybí pravá závorka a přitom byl problém zcela jinde.

3.3 Výběr frameworku

Při vypracovávání aplikace jsem použil pouze standardní JDK Javy [4], viz literatura [5] a [1] a čtyři externí knihovny.

První z nich je JDBC ovládač k MySQL databázi [5], který je potřeba, jak již název ovládače napovídá, k propojení Javy s MySQL databází. Druhou knihovnou, kterou jsem využil, je knihovna Jericho HTML parser [6], viz. literatura [7]. Tu jsem použil při načítání (parsování) informací ze serveru CSFD.cz. Předposlední knihovnou, kterou jsem použil při implementaci je JavaMail[7] viz literatura[6], pomocí které aplikace rozesílá varovné emaily těm známým, kteří nevrátili včas film. Poslední knihovnou, kterou jsem použil je knihovna JExcel viz literatura [9] a [10], pomocí které jsem naimplementoval export do xls souboru pro seznam všech evidovaných filmů.

Více o využití hledejte prosím v programátorské příručce, viz str. 36.

[4] Aktuální JDK Javy můžete stáhnout zde: http://java.sun.com/javase/downloads/widget/jdk6.jsp
[5] Aktuální verzi ovládače můžete stáhnout zde: http://dev.mysql.com/downloads/connector/j/
[6] Aktuální verzi knihovny můžete stáhnout zde: http://jericho.htmlparser.net/docs/index.html
[7] Aktuální verzi knihovny můžete stáhnout zde: http://java.sun.com/products/javamail/

4 Analýza a návrh vlastního řešení. Přístup ke zdrojům dat.

Analýza se skládá ze 4 částí:

- **Zadání** - v této části je uvedeno co se bude evidovat, kdo bude s programem pracovat (tzv. aktéři) a jaké budou vstupy a výstupy programu.
- **Datová analýza** - tato část se zabývá především ER diagramem, lineárním zápisem typů entit a vazbami mezi nimi, a také datovým slovníkem. Datová analýza se tak zabývá **ukládáním dat do databáze**, nejprve jsou nadefinovány jednotlivé tabulky, a poté i vazební vztahy mezi těmito tabulkami.
- **Funkční analýza** - v této části kapitoly se čtenář seznámí s kontextovým diagramem aplikace a s několika DFD diagramy. Grafickou formou tak jsou tak ukázány základní funkce systému.
- **Časová analýza** - v poslední části kapitoly je čtenář seznámen s časovou analýzou dvou entit - Film a Uživatel. Časová analýza definuje jednotlivé možné stavy dvou náhodně vybraných entit.

4.1 Zadání

Co budeme evidovat ?

- film – v této tabulce budeme evidovat všechny filmové média, které má daný uživatel
- vypůjčky - v této tabulce budeme evidovat všechny vypůjčky
- uživatelé – zde se budou evidovat všichni uživatelé tohoto programu
- známé – zde se budou evidovat všichni známí, kterým lze půjčit nějaké médium
- aktivní uživatel – zde se bude ukládat identifikátor aktuálně přihlášeného uživatele, pro jeho lepší dohledatelnost
- nastavení - zde se bude uchovávat nastavení aplikace
- email - zde se budou uchovávat přihlašovací údaje k SMTP serveru, pomocí kterého se budou odesílat varovné emaily těm, kteří nevrátí film včas.
- základní obrázky - zde se budou uchovávat základní obrázky. V současné chvíli je uložen obal pro film, kteérmu nebyl určen obal jiný.

Kdo?
S programem budou pracovat tři typy uživatelů:

- Administrátor - může vše, čili mazat uživatelé, povyšovat je na administrátory, obnovovat ztracené hesla, mazat databáze a podobně.

- Uživatel - může měnit vlastní nastavení, může používat vlastní databázi či sdílenou, nemůže mazat databáze, pouze mazat filmy ve své či sdílené databázi.
- Nepřihlášený uživatel – bude se moct jen zaregistrovat.

Vstupy:

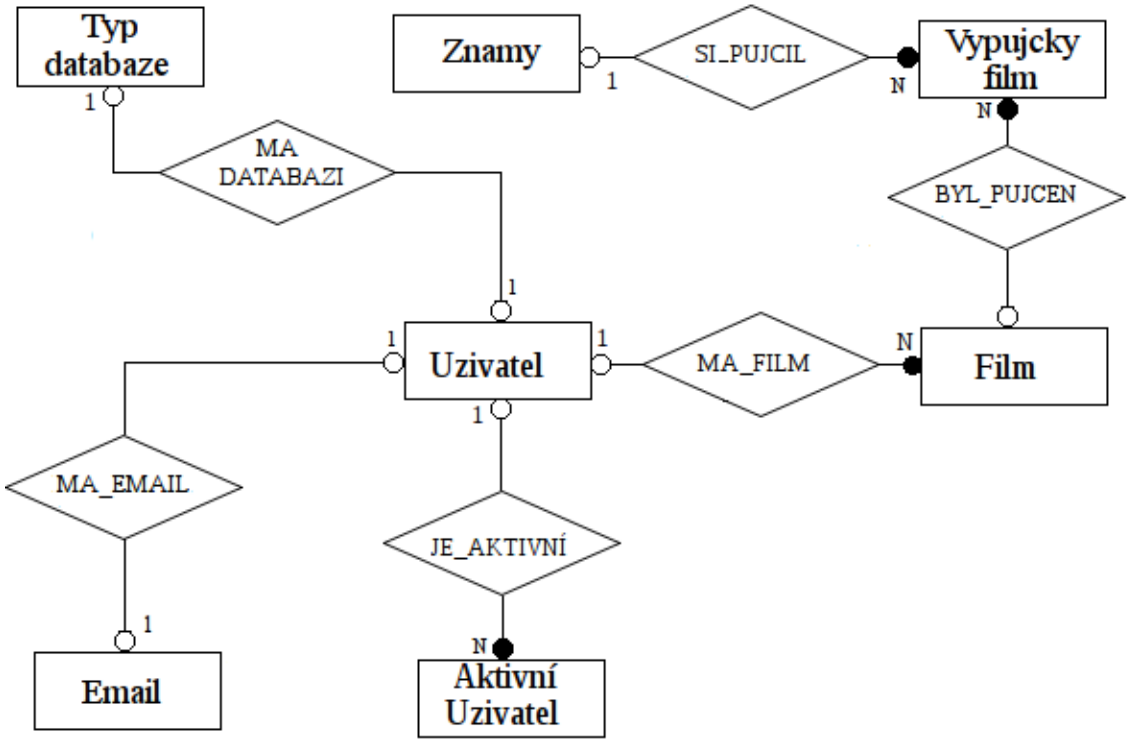
- U evidence filmů budeme evidovat základní informace (originální jméno, české jméno, rok výroby....), dále pak jeho hodnocení (uživatelské, čsfd, imdb), jazyky dabingů a titulků, odkaz na csfd.cz, imdb.com, odkaz na stránku filmu, poznámky uživatele, screenshot obalu, počet kusů médií, bonusy.
- U evidence vypůjček budeme evidovat id vypůjčky, kdo si film půjčil, který film si půjčil, kdy si ho půjčil, dokdy, zda-li byl upomínan, zda-li film byl již vrácen, a který uživatel film půjčil.
- U evidence uživatelů budeme evidovat jejich přezdívku, heslo, práva a nastavení.
- U evidence přátel budeme evidovat základní informace (jméno, adresa, email).
- V evidenci emailů budeme evidovat přihlašovací údaje k emailové schránce, pomocí které chceme posílat varovné emaily těm, kteří včas nevrátili film. V databázi se tedy budou uchovávat přístupové jméno a heslo k SMTP serveru, dále pak předmět a text varovných emailů, kódování emailu, port, pomocí kterého se budou emaily posílat, také zda-li SMTP server vyžaduje SSL protokol. Všechny tyto informace potřebuje i klasický emailový klient (Outlook, Thunderbird a podobně)
- V tabulce Nastavení budeme uchovávat nastavení aplikace. Nyní to tedy je půjčovní doba a cesta k internetovému prohlížeči.
- Tabulka Základní obrázky uchovává několik základních obrázků obalů DVD. V současné chvíli pouze ten, který informuje o tom, že uživatel nenačetl obal svůj vlastní.
- Typ databáze pak obsahuje ID privátních databází a loginy uživatelů, kterým patří tyto id.
- V tabulce aktivní uživatel bude identifikátor právě přihlášené uživatele, náhodně vygenerované id připojení a čas přihlášení a odhlášení.

Výstupy:

- Statistiky – textové
- Seznamy – seznam všech filmů

4.2 Datová analýza

4.2.1 ERD



Obrázek 4: ERD

4.2.2 Lineární zápis typů entit

Film - (id_filmu, jmeno_f,originalni_nazev_f,zanr_f, hodnoceni_csfd, hodnoceni_imdb, hodnoceni_vlastni_f,web_f,rezie_f, hraji_f, poznamky_f, typ_media_f, zvuk_f, titulky_f,obal_f, pocet_medii_f,rok_f, *login* (id uživatele, který vlastní film), *id_databaze* (id privátní databáze uživatele), bonusy_f,o_filmu)

Vypujcky_film - (ID_vypujcky,*kdo*,*co*,odkdy,dokdy, upominan,vraceno,*kdo_pujcil*)

Uzivatel - (login, jmeno_u, prijmeni_u, email_u, prava,sdilena_databaze,privatni_databaze, heslo, ulice_u, mesto_u, cislo_popisne, psc_u, zablokovany_u)

Znamy - (id_znamy_z, jmeno_z, prijmeni_z, ulice_z, cislo_z, mesto_z, psc_z, email_z)

Email - (*login*, host, od,pro,uzivatelske_jmeno, predmet, text_emailu, heslo, kodovani, port, zabezpeceni_ssl)

Nastaveni - (id_nastaveni, pujcovni_doba, prohlizec (cesta k prohlížeči))

Prihlaseny_uzivatel - (login, id_prihlaseni, cas_prihlaseni)

Typ_databaze - (ID, Typ_databaze) - ID = id privátní databáze, Typ databáze = login uživatele, kterému patří dané ID

Zakladni_obrazky - (id_obrazek, obrazek)

4.2.3 Lineární zápis typů vyzeb

SI_PUJCIL (Znamy,Vypujcky_film) - 1:N
BYL_PUJCEN (Vypujcky_film, Film) - N:1

MA_FILM (Uzivatel, Film) - 1:N
JE_AKTIVNI (Uzivatel, Aktivni_uzivatel) - 1:N

MA_EMAIL (Uzivatel,Email) - 1:1
MA_DATABAZI (Uzivatel, Typ_databaze) - 1:1

4.2.4 Datový slovník

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id_filmu	int	10	ANO	ANO	NE	primární klíč - unikátní položka
jmeno_f	varchar	255	NE	ANO	NE	žádné
originalni_nazev_f	varchar	255	NE	ANO	NE	žádné
zanr_f	varchar	255	NE	NE	ANO	zadny
hodnoceni_csfd	int	3	NE	NE	ANO	do 100
hodnoceni_imdb	int	2	NE	NE	ANO	do 10
hodnoceni_vlastni	int	3	NE	NE	ANO	libovolné číslo
web_f	varchar	255	NE	NE	ANO	webová adresa
rezie_f	varchar	255	NE	NE	ANO	žádný
hraji_f	varchar	1024	NE	NE	ANO	žádný
poznamky_f	varchar	1024	NE	NE	ANO	žádný
typ_media_f	varchar	255	NE	NE	ANO	žádný
zvuk_f	varchar	255	NE	NE	ANO	žádný
titulky_f	varchar	255	NE	NE	ANO	žádný
obal_f	longblob	-	NE	NE	ANO	obal DVD
pocet_medii_f	int	2	NE	NE	ANO	žádný
rok_f	int	4	NE	ANO	NE	žádný
login	varchar	255	NE	ANO	NE	cizí klíč z tabulky Uzivatele
id_databaze	int	6	NE	ANO	NE	cizí klíč z tabulky Typ_databaze
bonusy_f	text	-	NE	NE	ANO	žádný
o_filmu	text	-	NE	NE	ANO	žádný

Tabulka 1: Film

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id_vypujcky	int	11	ANO	ANO	NE	primární klíč - unikátní položka
kdo	int	10	NE	ANO	NE	žádné
co	int	11	NE	ANO	NE	žádné
odkdy	date	-	NE	NE	NE	žádné
dokdy	date	-	NE	NE	NE	žádné
upominan	varchar	255	NE	NE	ANO	ANO nebo NE
vraceno	varchar	-	NE	ANO	NE	ANO nebo NE
kdo_pujcil	varchar	255	NE	ANO	NE	cizí klíč - Uzivatel

Tabulka 2: Vypujcky_film

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
login	varchar	255	ANO	ANO	NE	primární klíč
jmeno_u	varchar	255	NE	ANO	NE	žádné
prijmeni_u	varchar	255	NE	ANO	NE	žádné
email_u	varchar	255	NE	NE	NE	žádné
prava	varchar	255	NE	NE	NE	Administrator nebo Uzivatel
sdilena_databaze	int	1	NE	NE	NE	ANO nebo NE
privantni_databaze	intr	1	NE	NE	NE	ANO nebo NE
ulice_u	varchar	255	NE	NE	ANO	žádné
cislo_u	varchar	255	NE	NE	ANO	žádné
mesto_u	varchar	255	NE	NE	ANO	žádné
psc_u	varchar	255	NE	ANO	ANO	5 číslic
zablokovany_u	int	1	NE	ANO	NE	žádné
heslo	varchar	255	NE	ANO	NE	žádné

Tabulka 3: Uzivatel

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id_znamy_z	int	10	ANO	ANO	NE	primární klíč
jmeno_z	varchar	255	NE	ANO	NE	žádné
prijmeni_z	varchar	255	NE	ANO	NE	žádné
ulice_z	varchar	255	NE	NE	ANO	žádné
cislo_z	varchar	255	NE	NE	ANO	žádné
mesto_z	varchar	255	NE	NE	ANO	žádné
psc_z	varchar	255	NE	ANO	ANO	5 číslic

Tabulka 4: Znamy

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
login	varchar	255	ANO	ANO	NE	primární klíč
host	varchar	255	NE	ANO	NE	žádné
od	varchar	255	NE	ANO	NE	email
pro	varchar	255	NE	NE	NE	email
uzivatelske_jmeno	varchar	255	NE	NE	NE	žádné
predmet	varchar	255	NE	NE	NE	žádné
text_emailu	text	-	NE	ANO	NE	žádné
heslo	varchar	255	NE	ANO	NE	žádné
kodovani	varchar	255	NE	ANO	NE	žádné
port	int	10	NE	ANO	NE	žádné
zabezpeceni_ssl	int	1	NE	ANO	NE	žádné

Tabulka 5: Email

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id_obrazek	int	4	ANO	ANO	NE	primární klíč
obrazek	longblob	-	NE	NE	NE	obrazek

Tabulka 6: Zakladni_obrazky

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id_nastaveni	int	3	ANO	ANO	NE	primární klíč
pujcovni_doba	int	2	ANO	ANO	NE	žádné
prohlizec	int	2000	ANO	ANO	NE	cesta k prohlížeči

Tabulka 7: Nastaveni

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
id	int	6	ANO	ANO	NE	primární klíč
typ_databaze	varchar	255	ANO	ANO	NE	žádné

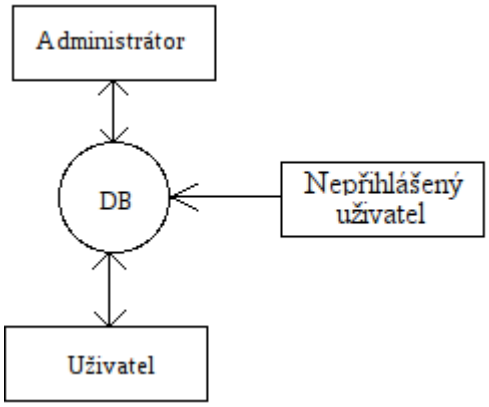
Tabulka 8: Typ_databaze

Název	Typ	Velikost	Klíč	Index	Null	IO, poznámky
login	varchar	255	ANO	ANO	NE	primární klíč,cizí klíč
id_prihlaseni	longblob	-	NE	NE	NE	náhodné číslo
cas_prihlaseni	datetime	-	NE	NE	NE	žádné
cas_odhlaseni	datetime	-	NE	NE	NE	žádné

Tabulka 9: Prihlaseny_uzivatel

4.3 Funčkní analýza

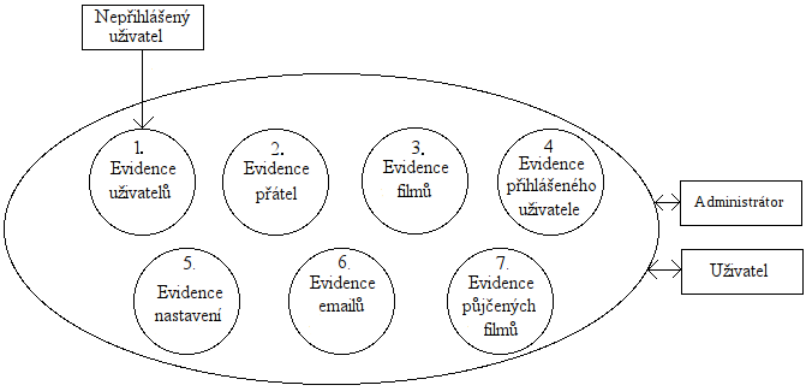
4.3.1 Kontextový diagram



Obrázek 5: Kontextový diagram aplikace

S aplikací budou pracovat tři aktéři: Administrátor, Uživatel, Neregistrovaný uživatel.

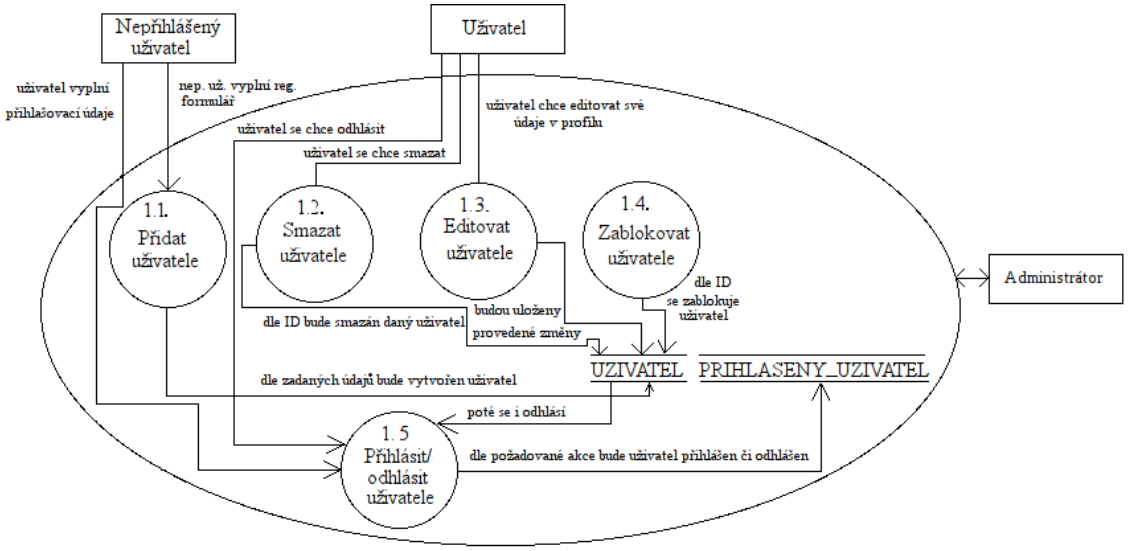
4.3.2 DFD 0.úrovně



Obrázek 6: DFD 0.úrovně

Administrátor i uživatel mohou přistupovat (ač uživatel s různými omezeními) ke všem tabulkám. Pouze nepřihlášený uživatel může pouze založit nový účet, čili přistupuje pouze k tabulce Uživatelé.

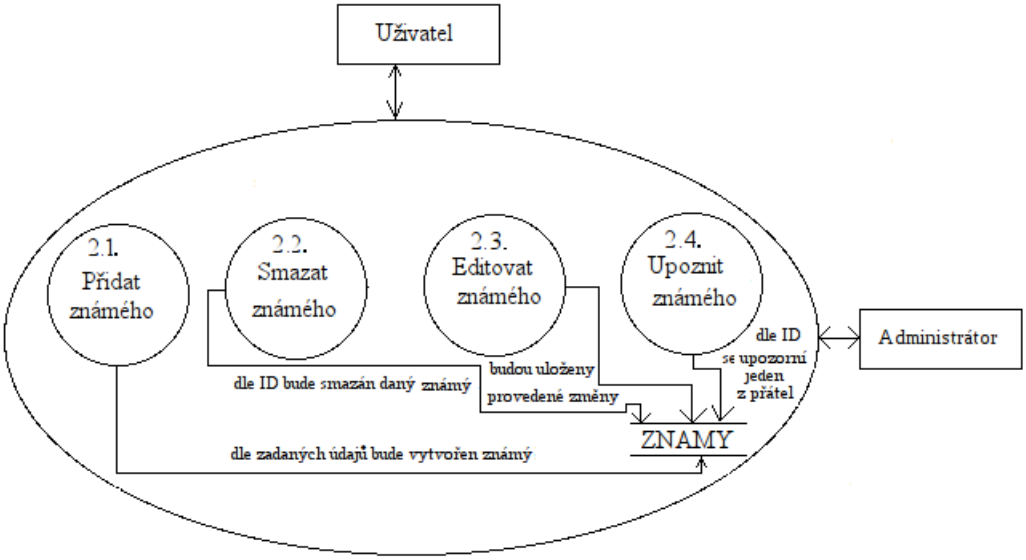
4.3.3 DFD 1.úrovně - správa uživatelů



Obrázek 7: DFD 1. úrovně

První úroveň DFD ukazuje, jaké činnosti jsou možné s jednotlivými uživateli. Nepřihlášený uživatel může pouze vytvářet nový profil. Klasický uživatel pak může měnit informace o sobě, ne však již pochopitelně o jiných uživatelích. Stejně tak může svůj profil smazat. Administrátor může cokoli.

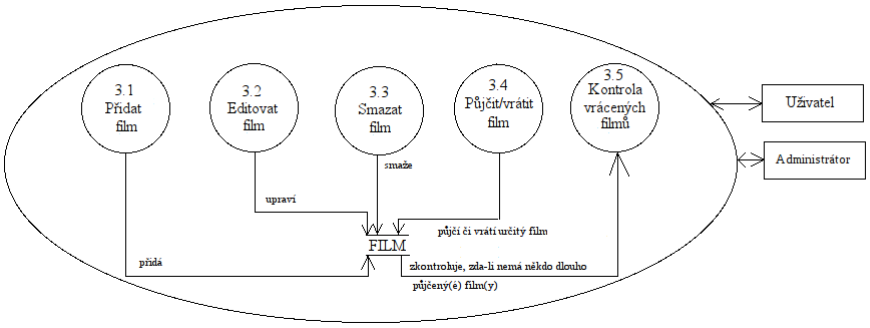
4.3.4 DFD 2.úrovně - správa známých



Obrázek 8: DFD 2. úrovně

Pokud se jedná o správu přátel (známých), čili těch, kterým je možné něco půjčit, pak zde se rozdíli mezi administrátorem a obyčejným uživatelem nerozlišují a oba aktéři mohou provádět stejné činnosti s databází známých (vytváření, editace, mazání).

4.3.5 DFD 3.úrovně - správa filmů

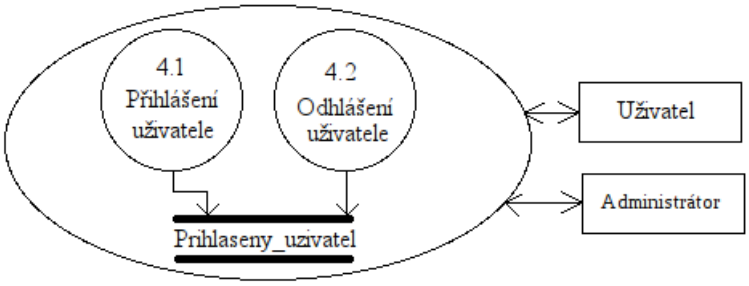


Obrázek 9: DFD 3. úrovně

Ani ve správě filmů není rozdílu mezi administrátorem a uživatelem, oba mohou totéž (vytvářet, editovat, půjčovat a vracet filmy). Aplikace sice řeší sdílené a privátní filmy, ale v DFD by toto řešila až patřičná podúroveň (3.1, 3.2, 3.3, 3.4, 3.5), tyto podúrovně

však již nejsou součstí práce, a to z důvodu hlavně místa, práce by se tak neúměrně „nafoukla“.

4.3.6 DFD 4.úrovně - Přihlašování

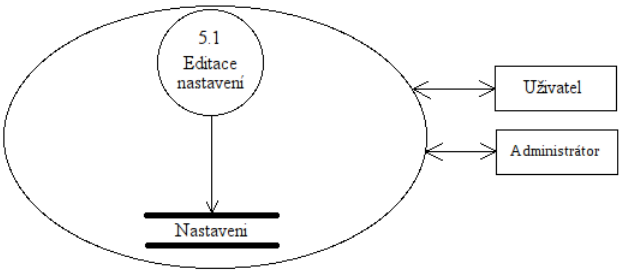


Obrázek 10: DFD 4. úrovně

Při přihlašování je každému uživateli určeno náhodné číslo, aby bylo umožněno jeho vícenásobné přihlášení, a taktéž bylo umožněno připojení více uživatelům najednou. Při přihlášení je zaznamenán čas a datum přihlášení.

Všechny údaje se načítají a ukládají do tabulky „Prihlaseny uzivatel“.

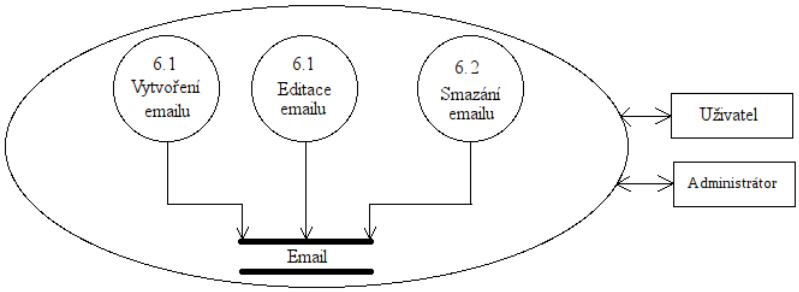
4.3.7 DFD 5.úrovně - Nastavení



Obrázek 11: DFD 5. úrovně

V databázi se také ukládá některá nastavení - půjčovní doba a cesta k prohlížeči. Tyto nastavení jsou určeny pro všechny uživatele. Nastavit je může kdokoli. Nastavení lze pouze modifikovat a po prvním spuštění aplikace je půjčovní doba nastavena na 30 dní a cesta k prohlížeči je prázdná. Nastavení se ukládá do stejnojmené tabulky.

4.3.8 DFD 6.úrovně - Emaily

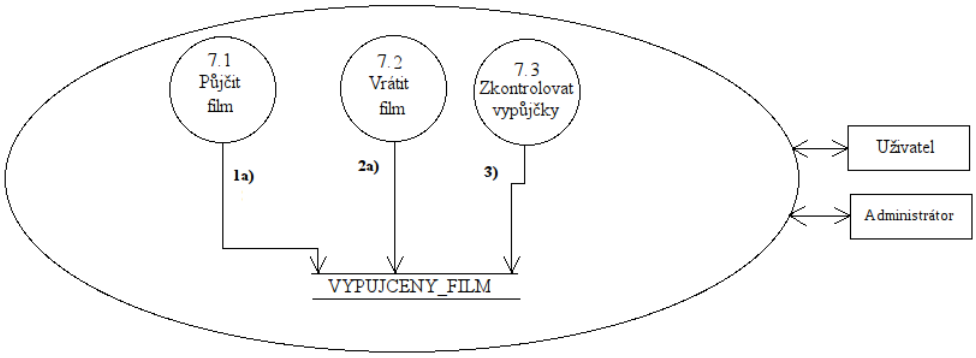


Obrázek 12: DFD 6. úrovně

Jak bylo již zmíněno v dřívější části práce, program umí posílat varovné emaily těm, kteří nevrátili film v určité lhůtě. Každý uživatel musí zadat přístupové údaje k emailu, pomocí kterého chtějí posílat upomínky. Tyto údaje pak uchovává tabulka „Email".

Každý uživatel má nastavení své, administrátor nastavení ostatních uživatelů nevidí a nemůže je měnit. Všechny údaje se načítají a ukládají do tabulky „Email".

4.3.9 DFD 7.úrovně - Vypůjčky



Obrázek 13: DFD 7. úrovně

Pro přehlednost jsem je vysvětlení procesů napsáno až pod obrázkem.

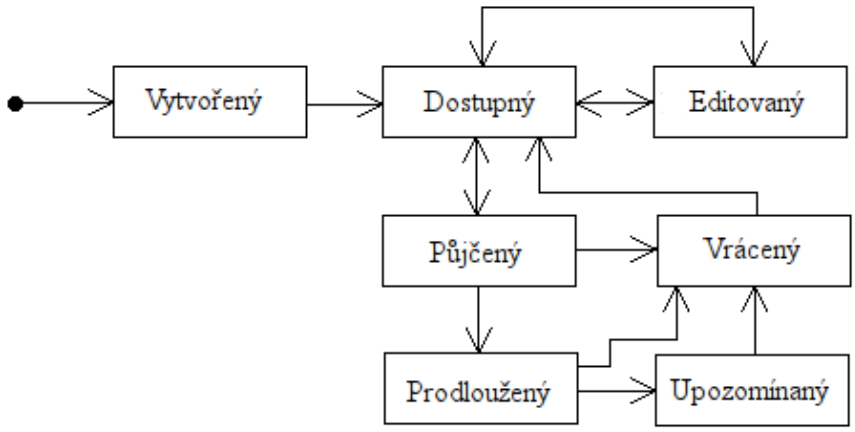
1) Proces zapíše do tabulky Pujceny film vše co je potřeba (za předpokladu, že je film k půjčení), tedy především ID daného půjčovaného filmu a ID známého, kterému jej půjčuji.

2) Proces vyhledá danou výpůjčku v tabulce Pujceny film, nastaví atribut vráceno na ANO.

3) Tento proces bude automatický a spustí se při každém spuštění aplikace. Jeho úkolem je zkontrolovat, zda-li některá výpůjčka v tabulce Pujceny film nepřesáhla nastavenou půjčovní dobu. Pokud ano, tak se zobrazí se hláška uživateli, který aplikaci spustil.

4.4 Časová analýza

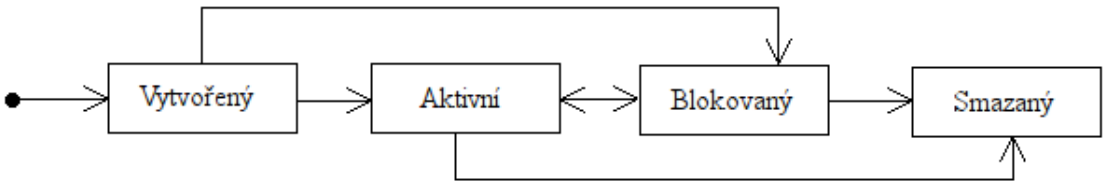
4.4.1 STD entity Film



Obrázek 14: STD entity film

Jakmile je určitá entita Film založena, pak ihned dostupná k půjčení, Jakmile si někdo film půjčí, film je označen jako půjčený. Pak může být buď vrácen, pokud vrácen není do určitého data, je film (i známý, který si ho půjčil) upomínaný. Jakmile je vrácen, je opět dostupný.

4.4.2 STD entity Uzivatel



Obrázek 15: STD entity film

Nový záznam typu Uživatel je vytvořený, poté se hned stává aktivním. Ihned po registraci má nově registrovaný uživatel obyčejná práva, výjimkou je situace, kdy se registruje první uživatel, ten je automaticky administrátorem. Uživatel může být taktéž zablokovaný Administrátorem, či jej Administrátor může kompletně zrušit či naopak povýšit na administrátora.

5 Návrh grafického rozhraní.

Grafické prostředí je vytvářeno, stejně tak i celá aplikace, pomocí IDE NetBeans, které se v aktuální chvíli nachází ve verzi 6.8. Pomocí tohoto IDE jsem tak pomocí metody Drag and Drop vytvořil celé GUI aplikace. Ruční implementace grafického prostředí by byla velmi náročná na čas, díky použití IDE byla tvorba GUI poměrně snadnou záležitostí. V API Javy jsou dostupné dvě kolekce grafických prvků:

- AWT - [8] starší, méně komponent, zastaralejší vzhled
- Swing [9]- modernější, více komponent, modernější vzhled

V dnešní době se již prakticky využívá balíček Swing, ani já tak nebyl při vývoji aplikace výjimkou, Look and Feel (soulad vzhledu aplikace se vzhledem konkrétního systému) jsem nemodifikoval a vzhled se tak vždy upraví dle konkrétního systému. Aplikace obsahuje celkem 6 hlavních oken:

- Přihlášení
- Hlavní okno se záložkami
- Nastavení přístupu k databázi
- Vytvoření nového uživatele
- Kontrola nevrácených filmů
- Nastavení odkazu na film

Základním kamenem aplikace jsou záložky, kdy každá z nich obsahuje určitou sadu informací (Filmy, Uživatelé, Známí, Nastavení a podobně). Při vývoji GUI aplikace jsem se částečně inspiroval u programu Ant movie catalog (záložka Filmy).

Při vývoji uživatelského prostředí jsem se snažil vytvořit co nejpřehlednější aplikaci, aby případný uživatel netápal, a pokud možno si věděl rady i bez manuálu. Proto jsem také, jak již zmiňuji v úvodu této kapitoly, použil záložky (komponenta TabbedPane z Java Swing), aby měl uživatel vše na očích. Naopak jsem se snažil, aby případný uživatel musel používat klasické menu co nejméně, z menu se tak provádějí méně časté činnosti, jako například exporty nebo vyhledávání filmů.

Aplikace obsahuje pouze českou lokalizaci. Java je však podporuje jak Internacionalizaci, tak i vlastní lokalizaci, proto by případný překlad aplikace neměl být problémem. Navíc celý proces značně urychlí i funkcionalita k tomu určená v NetBeans.

Pro přehlednost celého projektu jsou všechny třídy, starající se o vlastní GUI aplikace, umístěny ve společném balíčku (jmenuje se gui) a všechny třídy mají ve svém jméně společný prefix „gui“. Kromě kódu starající se o GUI aplikace a reakce na události aplikace,

[8] Více informací o Java AWT se dozvíte zde (EN): http://en.wikipedia.org/wiki/Abstract_Window_Toolkit
[9] Více informací o Java Swing se dozvíte zde (EN): http://en.wikipedia.org/wiki/Swing_(Java)

tyto třídy povětšinou neobsahují žádný stěžejní kód, ten je pouze volán pomocí událostí a nachází se v jiných třídách.

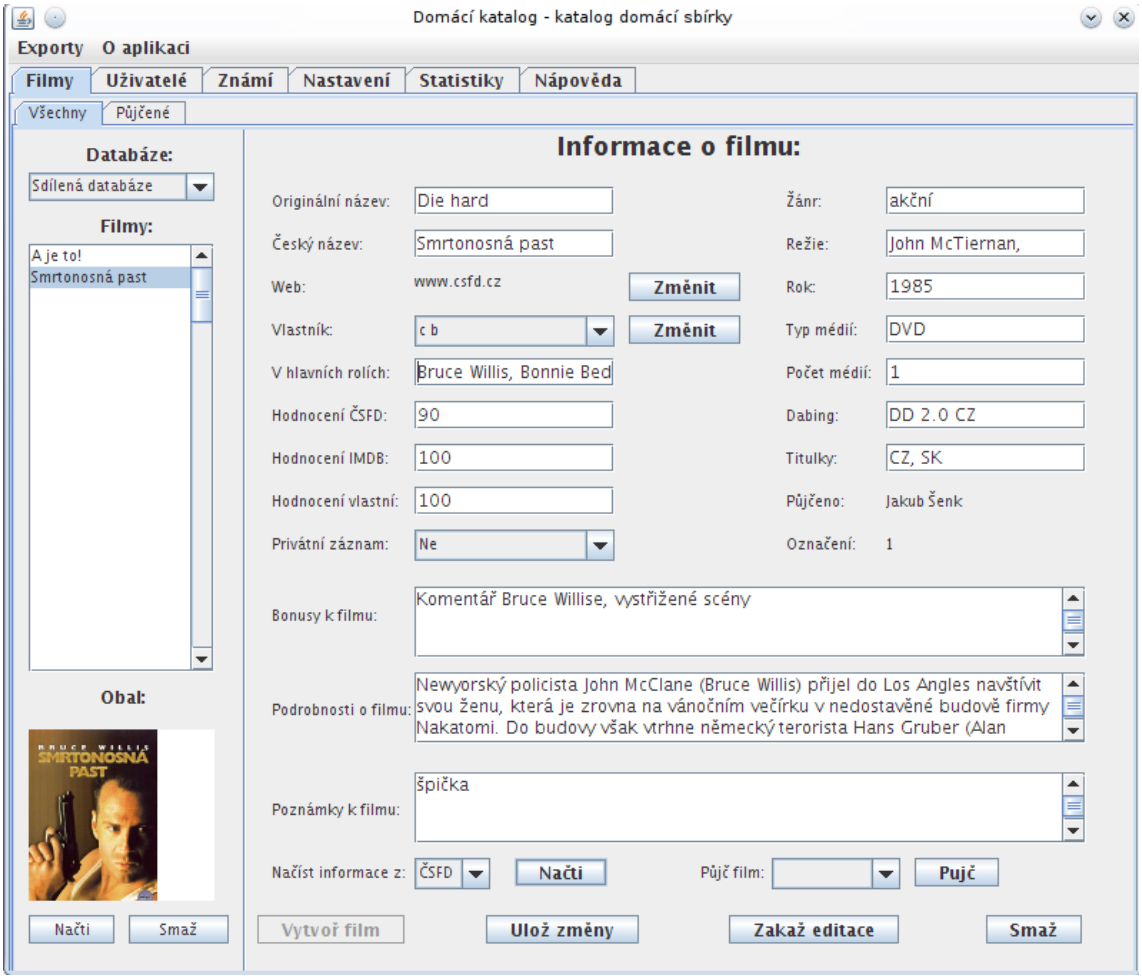
Některé komponenty jsou samozřejmě plněny daty až po spuštění aplikace a data se povětšinou načítají buď z databáze nebo ze serveru http://www.csfd.cz. K plnění dat povětšinou slouží opět další třídy.

5.1 Vzhled aplikace

Nyní pár slov k vlastnímu vzhledu. Pro ilustraci vzhledu aplikace přikládám dva screenshoty aplikace včetně jejich popisu, screenshoty byly vytvořeny v Mandrivě 2010 v grafickém prostředí KDE 4.4.2. Pro podrobnější seznámení s aplikací pak nalistujte uživatelskou příručku v této práci (str. 51).

Obrázek 16: Přihlášení do programu

První z nich ukazuje úvodní okno celé aplikace, které slouži k přihlašování do systému, tedy slouží k autentifikaci, díky které může poté dojít i k autorizaci (aplikace rozlišuje mezi administrátorem a obyčejným uživatelem). Krom vlastního přihlášení do programu lze i vytvářet nové uživatele (pokud zatím neexistuje žádný uživatel, vytvoří se administrátor, jinak se vytvoří pouze uživatel), či nastavit přihlašovací údaje k připojení do databáze.



Obrázek 17: Hlavní okno aplikace

Druhým ukázkovým oknem aplikace je hlavní okno aplikace, ve kterém se pracuje s databází filmů. V levé části okna si uživatel vybírá určitý film z určité databáze. Aplikace podporuje dva typy uživatelské databáze - sdílenou a privátní, obsah privátní databáze vidí pouze daný uživatel a jakýkoli administrátor. V pravé části okna se pak načítá detail daného filmu. Některé údaje o filmech lze načíst i z csfd.cz a v dalších verzích programu se počítá i s podporou dalších filmových internetových databank.

Podrobnější informace o ovládání programu se nacházejí v části „Uživatelská příručka“.

6 Realizace a programování.

6.1 Fáze realizace

Realizace bakalářské práce probíhala od června 2009 do května 2010. V počáteční fáze byly spíše teoretické a šlo nejprve o vlastní výběr tématu práce, vedoucího práce a poté nastínění základních rysů práce, čili jaké funkce by měl program podporovat a podobně. Poté byla vytvořena základní analýza, a to jak analýza konkurenčních programů, tak i analýza z pohledu vlastní aplikace.

V další fázi bylo vytvořeno základní GUI, kterému poté byla postupně přidávána funčnost, čímž byl nakonec vytvořen celý program. V poslední fázi vývoje byl výsledný program otestován, a také byl napsán text vlastní práce.

6.2 Vývojové prostředky

Při vývoji jsem použil několik programů. Prvním z nich bylo NetBeans IDE v aktuální verzi (v době realizace práce se jednalo o verzi 6.8). Jedná se o volně dostupné prostředí pro tvorbu nejen Java aplikací, které je pravděpodobně jedním ze dvou nejznámějších vývojových prostředí určených pro vývoj aplikací psaných v programovacím jazyce Java.

Pro správu MySQL databáze jsem použil tři nástroje.Prvním z nich byl MySQL Administrátor. Tento nástroj slouží ke správě MySQL databáze. Já osobně jej používal pouze pro zálohování databáze a k dalším účelům jsem používal konkurenční webovou aplikaci, která mne zaujala více. Touto webovou aplikací je aplikace phpMyAdmin, pomocí které jsem vytvářel jednotlivé tabulky, vazby a podobně. Občas jsem potřeboval otestovat určitý SQL dotaz, k těmto účelům jsem nejprve používal MySQL Query browser, který jsem později nahradil opět webovou aplikací phpMyAdmin, která tuto úlohu zastala stejně dobře.

6.3 Kód aplikace

Aplikace je pochopitelně předpřipravena právě pro NetBeans, po prvním načtení kódu aplikace v tomto IDE je pouze potřeba připojit pět JAR balíčků, které jsou potřeba pro správný chod programu. Prvním z nich je JDBC ovládač k MySQL databázi, který je potřeba pro správnou komunikaci mezi MySQL serverem a Javou. Druhým pak je balíček Jericho HTML parser pomocí kterého aplikace načítá data z csfd. Předposledním balíčkem, který je nutné připojit je balíček JExcel, pomocí kterého jsem naimplementoval podporu exportu pro seznam evidovaných filmů. Posledním vyžadovaným JAR balíčkem je mail.jar pomocí kterého aplikace posílá emaily, ze stejného důvodu je potřeba i balíček activation.jar. Žádné další balíčky aplikace nepotřebuje, protože jinak využívá standardní API k Javě SE. Všechny potřebné soubory najdete na přiloženém DVD k bakalářské práci a najdete v adresáři /src/domácí katalog/externí knihovny. Vlastní přidání dalších balíčků se provádí pomocí nastavení projektu. Výsledný JAR balíček s aplikací má již všechny potřebné soubory nalinkovány a načtou se sami po startu programu.

Pokud je vše provedeno správně, nevyskytnou se při otevírání projektu v NetBeans žádné problémy. Aplikace se skládá z 5 balíčků:

- Databáze
- Externí knihovny
- Gui
- Hlavni
- Třídy

V balíčku Databáze jsou třídy, starající se o práci s databází. První třída (MySQL.java) nejprve načte informace o MySQL serveru, ke kterému se má aplikace připojit a poté provede vlastní připojení. Informace o MySQL serveru, ke kterému se má aplikace připojit se nacházejí v TXT souboru umístěném v konfiguračním adresáři, který se nachází v domovském adresáři daného uživatele. Druhá třída (akce_s_db) obsahuje jednak implementaci 4 nejdůležitějších typů SQL dotazů (SELECT, INSERT, UPDATE, DELETE) ve formě statických metod, kterou jsou tak k dispozici každé třídě v projektu. Třída také provádí přihlášení do vlastní aplikace - provádí autentifikaci. Poslední třídou v balíčku je třída „ caste_sql_dotazy“, která je „zásobníkem“ často používaných SQL dotazů.

Balíček Externí knihovny obsahuje pouze JAR balíčky, které jsou potřeba pro korektní chod aplikace. Samozřejmě se zde nacházejí jen ty balíčky, které nejsou součástí API Java SE, čili ty, které jsem vyjmenoval výše.

Balíček Gui, jak již jeho název napovídá, obsahuje třídy, které mají za úkol vytvářet grafické prostředí aplikace. Každé jednotlivé samostatné okno aplikace má svůj vlastní soubor. Jelikož aplikace má 9 samostatných oken, i tento balíček tak má 9 java tříd.

Balíček Hlavní obsahuje opět tři třídy. První z nich (Ikona) se stará o implementaci klasické ikony aplikace vedle hodin. Tato ikona má pouze informativní charakter. Druhá (Main) je třídou hlavní, spouštěcí. Ihned po spuštění aplikace zobrazí hlavní okno. tato třída také obsahuje statické metody pro zobrazování a skrývání jednotlivých dalších oken.

Nejdůležitějším balíčkem je balíček Třídy, obsahující třídy, na jejichž základě se vytvářejí objekty, obsahující informace o určitém filmu, uživateli a podobně. Některé podstatné části z těchto tříd si popíšeme níže v této části práce.

6.4 Vybrané partie zdrojového kódu

6.4.1 Třída Film

Jednou z nejdůležitějších tříd je třída Film. Tato třída se stará o načítání, editování a další činnosti s jednotlivými filmy. Třídní proměnné pak slouží k načítání jednotlivých atributů z tabulky Film, a ke všem atributům jsou taky vytvořeny metody set a get, aby bylo možné je později editovat.

6.4.1.1 Načtení určitého filmu Určitý film z databáze se načítá pomocí konstruktoru třídy, který je uveden níže.

```
public Film(int id) throws SQLException
    {

        dotaz = "SELECT * FROM Film WHERE id_filmu = '"+id+"'";
        ResultSet r = databaze.akce_s_db.proved_select(dotaz);

        while (r.next())
        {
           id_filmu = r.getInt("id_filmu");
           jmeno_f = r.getString("jmeno_f");
           originalni_nazev_f = r.getString("originalni_nazev_f");
           zanr_f = r.getString("zanr_f");
           hodnoceni_csfd = r.getInt("hodnoceni_csfd");
           hodnoceni_imdb = r.getInt("hodnoceni_imdb");
           hodnoceni_vlastni_f = r.getInt("hodnoceni_vlastni_f");
           web_f = r.getString("web_f");
           rezie_f = r.getString("rezie_f");
           hraji_f = r.getString("hraji_f");
           poznamky_f = r.getString("poznamky_f");
           typ_media_f = r.getString("typ_media_f");
           zvuk_f = r.getString("zvuk_f");
           titulky_f = r.getString("titulky_f");
           pocet_medii_f = r.getInt("pocet_medii_f");
           obal_f = r.getBlob("obal_f");
           rok_f = r.getInt("rok_f");
           typ_media = r.getString("typ_media_f");
           vlastnik_f = r.getString("login");
           bonusy = r.getString("bonusy_f");
           o_filmu = r.getString("o_filmu");
           privatni_f = r.getInt("id_databaze");
        }

    }
```

Výpis 1: Konstruktor třídy Film

Konstruktor potřebuje znát ID filmu, který má načíst, toto id předáváme klasickou cestou - pomocí argumentu konstruktoru. Jakmile tedy vytvoříme objekt a nepoužijeme konstruktor výchozí, ale přetížený a jako argument použijeme id požadovaného filmu, tak konstruktor si nejprve pomocí SQL dotazu načte do ResultSetu daný film a poté načtené jednotlivé atributy u daného filmu uloží do třídních proměnných, které se následně zobrazí ve formuláři, o což se stará již jiná třídní metoda.Konstruktory jsou v této třídě celkem tři - výchozí, výše uvedený a poslední, jehož argumentem je celý ResultSet.

Další důležitou funkcí, která je již zmíněna výše, je načtení filmu z databáze. O to se stará výše uvedený konstruktor, informace o filmech jsou zatím pouze v proměnných a nikde se tedy zatím nezobrazují. O toto se stará třídní metoda „nacti_film_ do formulare“.

```
public void nacti_film_do_formulare() throws SQLException
    {
      try
      {
        //nejprve nacteme obal filmu
        hlavni.podpurne_funkce.zobraz_obal_filmu(this.obal_f.getBinaryStream());
      } catch (IOException ex)

      {
          Logger.getLogger(Film.class.getName()).log(Level.SEVERE, null, ex);
      }

      //pote zobrazime informace o urcitem filmu v formulari aplikace
      //k objektu pristupujeme pomoci klicoveho slova this

      gui.gui_main.originalni_nazev_f.setText(this.getOriginalni_nazev_f());
      gui.gui_main.jmeno_f.setText(this.getJmeno_f());
      gui.gui_main.hraji_f.setText(this.getHraji_f());
      gui.gui_main.hraji_f.setCaretPosition(0);
      gui.gui_main.web_f.setText(this.getWeb_f());
      gui.gui_main.hodnoceni_vlastni_f.setText(Integer.toString(this.getHodnoceni_vlastni_f()));
      gui.gui_main.hodnoceni_csfd.setText(Integer.toString(this.getHodnoceni_csfd()));
      gui.gui_main.hodnoceni_imdb.setText(Integer.toString(this.getHodnoceni_imdb()));
      this.nacti_mozne_vlastniky();
      gui.gui_main.bonusy_f.setText(this.getBonusy());
      gui.gui_main.bonusy_f.setCaretPosition(0);
      gui.gui_main.o_filmu.setText(this.getO_filmu());
      gui_main.o_filmu.setCaretPosition(0);
      gui.gui_main.zanr_f.setText(this.getZanr_f());
      gui.gui_main.rezie_f.setText(this.getRezie_f());
      gui.gui_main.pocet_medii_f.setText(Integer.toString(this.getPocet_medii_f()));
      gui.gui_main.zvuk_f.setText(this.getZvuk_f());
      gui.gui_main.titulky_f.setText(this.getTitulky_f());
      gui.gui_main.poznamky_f.setText(this.getPoznamky_f());
      gui.gui_main.poznamky_f.setCaretPosition(0);
      gui.gui_main.rok_f.setText(Integer.toString(this.getRok_f()));
      gui.gui_main.typ_media_f.setText(this.getTyp_media_f());
      gui.gui_main.oznaceni_filmu.setText(Integer.toString(this.getId_filmu()));
      gui.gui_main.pujceno.setText(this.kdo_ma_film_pujcen());
      int privatni = this.getPrivatni_f();

      if (privatni == 1) {
          gui.gui_main.privatni.setSelectedIndex(1);
      } else {
          gui.gui_main.privatni.setSelectedIndex(0);
      }
       }
```

Výpis 2: Načtení filmu do formuláře

Máme tedy již vytvořený objekt Film pomocí přetíženého konstruktoru, kterému jsme předali v arumentu id filmu, který chcme načíst. Informace o filmu jsou načteny v proměnných objektu, nyní stačí tato data zobrazit ve formuláři, o což se stará výše uvedená třídní metoda. Nejprve je načten obrázek, k čemuž slouží další funkce. Poté je již do každé komponenty načten patřičný obsah z dříve vytvořeného objektu. Nakonec je nutné zjistit, zda je film privátní či nikoli a dle toho tuto informaci zobrazit v patřičné komponentě.

Celý proces načtení určitého filmu a jeho zobrazení v programu je prováděno takto:

```
Film x = new Film(r.getInt("id_filmu"));
x.nacti_film_do_Formulare();
```

Výpis 3: Načítáme film

Nejprve se vytvoří nový objekt třídy Film. Díky použití přetíženého konstruktoru bude ihned obsahovat infromace o daném filmu. V druhém kroku se jen načtené informace zobrazí.

6.4.1.2 Editace určitého filmu Editace filmu je prováděna na podobném principu, nejprve se vytvoří prázdný objekt Film, poté se načte obsah jednotlivých komponent (čili toho, co uživatel vepsal do patřičných „kolonek“ ve formuláři) a nakonec je daný film modifikován pomocí standardního SQL dotazu UPDATE.

```
public void nacti_data_z_formulare() throws SQLException
    {
        jmeno_f = gui.gui_main.jmeno_f.getText();
        originalni_nazev_f = gui.gui_main.originalni_nazev_f.getText();
        zanr_f = gui.gui_main.zanr_f.getText();
        hodnoceni_csfd = Integer.parseInt(gui.gui_main.hodnoceni_csfd.getText());
        hodnoceni_imdb = Integer.parseInt(gui.gui_main.hodnoceni_imdb.getText());
        hodnoceni_vlastni_f = Integer.parseInt(gui.gui_main.hodnoceni_vlastni_f.getText());
        web_f = gui.gui_main.web_f.getText();
        rezie_f = gui.gui_main.rezie_f.getText();
        hraji_f = gui.gui_main.hraji_f.getText();
        poznamky_f = gui.gui_main.poznamky_f.getText();
        typ_media_f = gui.gui_main.typ_media_f.getText();
        zvuk_f = gui.gui_main.zvuk_f.getText();
        titulky_f = gui.gui_main.titulky_f.getText();
        pocet_medii_f = Integer.parseInt(gui.gui_main.pocet_medii_f.getText());
        rok_f = Integer.parseInt(gui.gui_main.rok_f.getText());
        typ_media = gui.gui_main.typ_media_f.getText();
        bonusy = gui.gui_main.bonusy_f.getText();
        o_filmu = gui.gui_main.o_filmu.getText();

        //nyni musime zjistit, zdali ma byt po editace zaznam privatni ci nikoli
        int index = -1;

        index = gui.gui_main.privatni.getSelectedIndex();

        if (index != 0)
        {
            //dany zaznam je nadale sdileny
            privatni_f = 1;


        }
        else
            if (index == -1)
            {
                //teoreticka chyba, ktera muze nastat
                //v tom pripade nastavime dany film jako sdileny
                privatni_f = 1;

            }
            else

            {
                //pokud je privatni

                //nejprve si musime zjistit id dane soukrome skupiny pro
                //prihlaseneho uzivatele
```

```
                    //nejprve si musim zjistit kdo je aktualne prihlasen
                    String prihlasen = Uzivatel.getLoginPrihlasenehoUzivatele();

                    //nyni musime zjistit ID jeho skupiny
                    int ID = Uzivatel.getIDPrivatniDatabazeUzivatele(prihlasen);

                    privatni_f = ID;
                }
    }
```

Výpis 4: Načtení obsahu komponent pro uložení změněného filmu

Nyní jsou tedy všechny údaje o filmu načteny v přepřipraveném objektu. Nyní stačí tedy vytvořit a provést SQL dotaz UPDATE, čímž změny uložíme.

```
    public void uloz_zmeny_do_db()
        {

          String dotaz = "UPDATE␣Film␣SET␣"+

           "jmeno_f='"+jmeno_f+"'␣,␣"+
           "originalni_nazev_f='"+originalni_nazev_f+"'␣,␣"+
           "zanr_f='"+zanr_f+"'␣,␣"+
           "hodnoceni_csfd="+hodnoceni_csfd+"␣,␣"+
           "hodnoceni_imdb="+hodnoceni_imdb+"␣,␣"+
           "hodnoceni_vlastni_f="+hodnoceni_vlastni_f+"␣,␣"+
           "web_f='"+web_f+"'␣,␣"+
           "rezie_f='"+rezie_f+"'␣,␣"+
           " hraji_f ='"+ hraji_f +"'␣,␣"+
           "poznamky_f='"+poznamky_f+"'␣,␣"+
           "typ_media_f='"+typ_media_f+"'␣,␣"+
           "zvuk_f='"+zvuk_f+"'␣,␣"+
           " titulky_f ='"+ titulky_f +"'␣,␣"+
           "pocet_medii_f="+pocet_medii_f+"␣,␣"+
           "id_databaze="+ privatni_f+"␣,␣"+
           "rok_f="+rok_f+"␣,␣"+
           "bonusy_f='"+bonusy+"'␣,␣"+
           "o_filmu='"+o_filmu+"'␣␣"+
           "WHERE␣id_filmu="+gui.gui_main.oznaceni_filmu.getText()+";";

         databaze.akce_s_db.proved_update(dotaz);

         System.out.println("\n\n␣Bude␣proveden␣tento␣MySQL␣dotaz"+dotaz);

        }
```

Výpis 5: Uložení změněného filmu

Vytváření SQL dotazu je poměrně jednoduché, stačí k vlastnímu SQL dotazu použít klasický datový typ String, do něj vepsat příslušný SQL dotaz a ten potom pomocí předpřipravých metod nechat vykonat. Změněné informace o filmu nesou jednotlivé proměnné objektu, stačí je tedy použit. Vlevo je vždy uvedeno jméno atributu, které chceme

změnit a vpravo za = je potom vlastní hodnota, kterou chceme u daného atributu daného záznamu uložit. Pro kontrolu SQL dotazu se tento dotaz vypisuje na standardním výstupu.

6.4.1.3 Mazání určitého filmu Vymazání určitého filmu z databáze je velmi jednoduché, stačí znát id filmu, které chceme smazat, potom stačí provést vlastní vymazání. Před vlastním smazáním filmu je volána funkce „je film pujcen“, která před vlastním smazáním filmu ověří, zda není náhodou film půjčen, v tom případě nedovolí jeho smazání.

```
public static void je_film_pujcen(int id_filmu) throws SQLException
  {

        String dotaz = "SELECT␣COUNT(kdo)␣FROM␣Vypujcky_film␣WHERE␣co="+id_filmu+"␣
            and␣vraceno='NE';";
        ResultSet r = akce_s_db.proved_select(dotaz);
        System.out.println(dotaz);

        int pocet = 0;

        while (r.next())
        {

            pocet = r.getInt(1);


        }

        if (pocet==0)
        {
            // film nen pujcen, lze smazat!
            //smaz(id_filmu);

            String dotaz2 = "DELETE␣FROM␣Film␣WHERE␣id_filmu␣=␣'"+id_filmu+"';";
            System.out.println(dotaz2);
            databaze.akce_s_db.proved_delete(dotaz2);

            podpurne_funkce.zobraz_oznameni("Film␣smazan");

        }

        else
        {

            podpurne_funkce.zobraz_chybu("Film␣nelze␣smazat␣-␣je␣pujcen!");

        }


  }
```

Výpis 6: Smazání daného filmu

Nejdříve tedy tato funkce, pomocí SQL dotazu, ověří, zda-li je daný film půjčen. SQL dotaz vrací dvě hodnoty, jedničku a nulu. Pokud vrátí jedničku, znamená to, že film je půjčen, protože existuje jeden člověk, co má film půjčen. Pokud vrátí číslo větší než 1, pak je něco v nepořádku, protože by to znamenalo, že daný konkrétní film mají půjčeni dva lidé, což se samozřejmě nemůže stát. Toto se tak může stát pouze v případě poruchy programu, či ručního zásahu přímo do databáze. V tomto případě se film označí za půjčený a nebude smazán.

Pokud je tedy zjištěno, že film je půjčen, bude tato skutečnost oznámena uživateli. V opačném případě dojde k vymazání filmu a i o tomto bude uživatel informován. Pro zobrazení chybových a informačních hlášek byly vytvořeny k tomu určené funkce - *zobraz chybu (String text)* a *zobraz upozorneni (String text)*.

6.4.2 Třída Vypujcky film

Jednou z velmi žádaných funkcí programů pro správu filmových (a případně i jiných) sbírek je půjčování. Jelikož se má aplikace specializuje na domácí sbírky, rozhodl jsem se, že půjčování udělám pouze základní s tím, že některé funkce (cena za půjčení, penále, atd..) nebudu implementovat. Program tak umí provádět vypůjčky, vracet je, upozorňovat na nevrácené filmy (jak uživatele, tak i toho, kdo si film půjčil) a samozřejmě lze i vypůjčky prodlužovat.

6.4.2.1 Vytvoření vypůjčky

O vytvoření vypůjčky, stejně tak o vše ostatní, co se týká vypůjček, se stará třída „Vypujcky_film“.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        int test = -1;

        if (jComboBox2.getSelectedIndex()==-1)
        {

            podpurne_funkce.zobraz_oznameni("Nevybral jste jmeno znameho, kteremu chcete
                pujcit film");

        }

        else
        {
            //jeste nez pujcime dany film, musime si overit, zda-li neni film jiz pujceny
            try {
                    Film x = new Film(Integer.parseInt(oznaceni_filmu.getText()));

                    String dotaz = "SELECT * FROM Vypujcky_film WHERE co="+Integer.
                        parseInt(oznaceni_filmu.getText())+" and vraceno='NE'";
                    ResultSet r2 = akce_s_db.proved_select(dotaz);
                    System.out.println(dotaz);

                    while (r2.next())
```

```
                {

                    test = r2.getInt("kdo");


                }

                if (test==-1)
                    {


                        System.out.println("Film byl pujcen");
                        x.pujc_film(jComboBox2.getSelectedIndex());
                        podpurne_funkce.zobraz_oznameni("Film pujcen!");


                    }
                    else
                    {

                        podpurne_funkce.zobraz_oznameni("Film je jiz pujcen!");
                        System.out.println("Film nelze pujcit!");

                    }

            } catch (SQLException ex)
            {
                Logger.getLogger(gui_main.class.getName()).log(Level.SEVERE, null, ex);
            }

        }
    }
```

Výpis 7: Vytvoření vypůjčky

V první podmínce se zjišťuje, zda-li byl vybrán určitý známý, komu je film půjčován. Pokud vybrán nebyl, uživatel o tom bude informován a vypůjčka samozřejmě vytvořena nebude. V opačném případě bude následně ověřeno, zda-li film již není půjčen. Opět, pokud bude toto zjištěno, vypůjčka nebude provedena a uživatel o tom bude informován. V opačném případě bude vypůjčka provedena a uživatel o tom bude také informován.

Vlastní vytvoření vypůjčky pak vykonává metoda pujc_film, která se nachází v třídě „vypujcky_film“.

```
public void pujc_film(int id_vyberu) throws SQLException
 {

    ResultSet r = caste_sql_dotazy.zjisti_vsechny_zname();
    int kontrola = 0;

    while (r.next())
```

```
        {

            if (kontrola==id_vyberu)
            {

                Vypujcky nova = new Vypujcky();

                nova.setCo(this.id_filmu);
                nova.setKdo(r.getInt("id_znamy_z"));
                nova.setKdo_pujcil(Uzivatel.getLoginPrihlasenehoUzivatele());

                nova.vytvor_vypujcku();

                kontrola++;
            }
            else
            {

                 kontrola++;

            }

        }


    }
```

Výpis 8: Vytvoření vypůjčky 2

Hned na začátku této metody se zjišťuje, koho uživatel vybral jako známého, komu chce daný film půjčit. Načteme si všechny známé a vybraného známého zjistíme tak, že si zjistíme index vybrané položky v jComboBoxu, pomocí kterého uživatel vybírá známého, kterému chce film půjčit a tento index pak říká pořadové číslo v seznamu všech dostupných známých. Jelikož nelze v ResultSetu přistupovat na určitou pozici, ale musíme vždy projít celý ResultSet, pak při načtení daného známého musíme porovnat jeho pořadové číslo s hledaným, a pokud se tyto dvě čísla rovnají, pak máme načteného známého, kterému se má film půjčit. Toto se provádí v prvních 4 řádcích kódu.

Následně, vytvoříme novou vypůjčku (opět se jedná o objekt třídy, tentokráte se tedy jedná o objekt třídy „Vypujcky_film“), do něj uložíme potřebné údaje a pomocí třídní funkce „vytvor_vypujcku“ vytvoříme vlastní vypůjčku. Opět se jedná o standardní SQL dotaz INSERT.

6.4.2.2 Prodloužení vypůjčky

Prodloužení vypůjčky probíhá na uživatelem definovanou délku. Nastavení se ukládá do databáze a je pro všechny uživatele stejné.

```
public static void prodluz_vypujcku(int id_vypujcky) throws SQLException
    {

        // toto je slozitejsi v tom, ze nejdrive musime zjistit, do jakeho data
        // vypujcku prodluzujeme a pote toto datum ulozit do DB
```

```
        //1. nejdrive musme zjistit , jaka je nastavena vypujcni doba
        int vypujcni_doba = Vypujcky.zjisti_pujcovni_dobu();

        //2. nyni zjistime datum, do ktereho prodluzujeme vypujcni dobu filmu
        String dotaz = "SELECT␣DATE_ADD(CURDATE(),␣INTERVAL␣"+vypujcni_doba+"␣DAY)␣
            AS␣VYSLEDEK␣FROM␣Vypujcky_film␣WHERE␣ID_vypujcky="+id_vypujcky+";";
        ResultSet r= akce_s_db.proved_select(dotaz);

        //datum mame zjisteno, staci jej aktualizovat v DB
        while (r.next())
        {

            String dotaz2 = "UPDATE␣Vypujcky_film␣SET␣"+
                            "dokdy␣=␣'"+r.getDate("VYSLEDEK")+
                            "'␣WHERE␣ID_vypujcky="+id_vypujcky+";";

            akce_s_db.proved_update(dotaz2);

            System.out.println(dotaz2);

        }


    }
```

Výpis 9: Prodloužení vypůjčky

Myslím si, že tento kód je dobře okomentován a jako takový nepotřebuje tedy dlouhé komentáře. Prodloužení vypůjčky je prováděno ve třech krocích. Nejprve se zjistí nastavená vypůjční doba. Poté se musí zjistit datum, do kterého se vypůjčka prodlouží. Počet dní, o které se vypůjčka prodlouží se připočítávají k aktuálnímu dni. Nakonec se provede vlastní prodloužení pomocí dotazu UPDATE.

6.4.2.3 Vrácení vypůjčky

Vrácení vypůjčky je jednoduchou funkcí, která jen změní atribut VRACENO v tabulce „Vypujcky film“ z NE na ANO.

```
public static void vrat_film(int id_vypujcky)
    {

        String dotaz = "UPDATE␣Vypujcky_film␣SET␣"+
                       "vraceno␣=␣'ANO'"+
                       "␣WHERE␣ID_vypujcky="+id_vypujcky+";";

         akce_s_db.proved_update(dotaz);

         podpurne_funkce.zobraz_oznameni("Film␣vrcen!");

    }
```

Výpis 10: Prodloužení vypůjčky

Provede se tedy pouze jeden UPDATE SQL dotaz, který provede výše uvedené.

6.4.3 Načítání z ČSFD

Jako doplňková funkce bylo naimplementováno načítání některých informací o filmu z československé filmové databáze, neboli csfd.cz. Původně bylo plánováno použití omezeného přímého přístupu do databáze, avšak takovýto přístup tento server nedovoluje, a to ani po písemném dotazu ve formě emailu adresovaného přímo vedení serveru.

Nakonec jsem využil klasického HTML parsování. API Javy však k tomuto neposkytuje dostatečné prostředky, proto jsem se rozhodl použít Jericho HTML parser, který je dostupný pod licencemi Eclipse Public License (EPL) a GNU Lesser General Public License (LGPL). ze stránek výrobce - http://jericho.htmlparser.net/.Stačilo tedy k projektu připojit JAR soubor obsahující API a implementace mohla započít.

Načítání z csfd se skládá ze dvou kroků - v prvním se hledá stránka konkrétního filmu, v druhém se poté tato stránka parsuje. K oběma krokům bylo vytvořeno GUI, které obsahuje vyjímečně i výkonný kód, jelikož se jedná pouze o dvě funkce.

6.4.3.1 Vyhledání stránky filmu

V prvním kroku je tedy potřeba vyhledat stránku daného filmu. Toto je jednodušší ze dvou výše uvedených kroků. Stačí tedy načíst stránku „http://csfd.cz/hledani-filmu-hercu-reziseru-ve-filmove-databazi/?search=“ + jméno filmu a tu následně rozparsovat a zjistit filmy, které byly vyhledány a seznam těchto filmů zobrazit uživateli k výběru.

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        try {

    1:  Source source = new Source(new URL("http://csfd.cz/hledani-filmu-hercu-reziseru-ve
          -filmove-databazi/?search=" + jTextField1.getText()));

        2:  int ukazatel=0;

        3:  source.fullSequentialParse();
        4:  List<Element> linkElements = source.getAllElements(HTMLElementName.A);

    5:  for (Element linkElement : linkElements)
     {

    6:      String href = linkElement.getAttributeValue("href");

    7:      if (href == null)
        {
          continue;
        }

    8:      String label = linkElement.getContent().getTextExtractor().toString();

    9:      if (label.startsWith(jTextField2.getText()))
     {
```

```
  a) pole_filmu[ukazatel]=label;
  b) pole_odkazu[ukazatel]=href;

  c) ukazatel++;
        }


    }
} catch (IOException ex) {

    java. util .logging.Logger.getLogger(gui_najdi_film.class.getName()).log(Level.
        SEVERE, null, ex);
}

 10:  jList1 .removeAll();
 11:  jList1 .setListData(pole_filmu);
```

Výpis 11: Načítání z CSFD - nalezení stránky filmu

Nejprve se načte požadovaná stránka do proměnné source, která má stejnojmený datový typ(řádek 1, dále jen číslo). Proměnná ukazatel (2) slouží jako ukazatel vrcholu polí, které používám, a sice „pole filmu“ (9a) obsahující jmenovky všech nalezených filmů a „pole odkazu“ (9b) obsahující odkazy na csfd stránky těchto filmů.

Poté je již spuštěno vlastní načtení celé WWW stránky (3), a poté jsou načteny všechny A tagy včetně všech jeho atributů (4). Následně pro každý tag (5) je načten atribut href (6), který obsahuje adresu na stránku daného filmu. Pokud je odkaz null (7), čili tento atribut u daného tagu neexistuje, dále se nic nevykonává a algoritmus skočí na další tag. Nyní se uloží text odkazu (8). Musíme načíst jen ty odkazy, které obsahují název filmu (9 a,b) (stránka totiž obsahuje i například reklamní odkazy a jiné, a také platí, že všechny odkazy obsahují jméno filmu, uživatel však musí zadat bezpodmínečně několik počátečních písmen názvu filmu, jinak nebude film nalezen).

Jakmile máme vše potřebné načteno, ikrementujeme ukazatel na vrchol pole (9c) a hledáme další odkaz. Poté jen zobrazíme výsledek hledání v jListu (11), nejdříve však musíme smazat předešlé hledání (10).

6.4.3.2 Pasování stránky filmu Druhý krok je již těžším - je nutné rozparsovat danou stránku filmu a tím získat potřebné údaje. Některé údaje, jako třeba rok výroby filmu, jsou však uvedeny v prázdných tag entitách, čili v takových, které nemají vůbec žádné atributy a nepřišel jsem na žádný způsob, jak je načíst, protože takových je na stránce spousta a nelze tak nijak definovat pravidlo, jak jej má parser jednoznačně poznat. Program tak umí načíst pouze české jméno filmu, herce, režiséry, informace o ději filmu a jeho hodnocení. Zbylé údaje tak musí uživatel vepsat sám. Níže uvedená ukázka kódu demonstruje načtení informace o ději filmu.

```
1) List<Element> linkElements_div = source.getAllElements(HTMLElementName.DIV);

   2) for (Element linkElement : linkElements_div) {

   3) String style = linkElement.getAttributeValue("style");

   4) if (style == null)
      {
          continue;
      }

   5) String label = linkElement.getContent().getTextExtractor().toString();


   6) if (style.compareTo("float:left;width:425px;padding-top:10px;font-weight:normal
      ")==0)
      {


          System.out.println(label);
   7)     gui_main.o_filmu.setText(gui_main.o_filmu.getText()+label+",␣");

      }



      }
```

Výpis 12: Načítání z CSFD - nalezení informací o filmu

Postup je obdobný, nejprve se načtou všechny tagy, tentokráte se načítají tagy DIV (bod 1, dále opět jen číslovka). Následně se pro každý načtený tag DIV (2) zjišťuje hodnota atributu STYLE (3), pokud je nulová (4), pak se automaticky skočí na další tag, jinak se načte text odkazu do proměnné (5). Pokud se atribut style = hledané hodnotě, pak jsme našli námi požadovaný tag, který jako jediný má style atribut o výše uvedené hodnotě. Nyní tedy stačí zobrazit informace o filmu k tomu určené komponentě (7).

Výše uvedený přístup má jednu zásadní nevýhodu - jestliže tvůrci webu změní některé tagy, je potřeba výše uvedené předělat. Bohužel jsem nenašel lepší způsob načtení informací z jejich stránek.

7 Uživatelská dokumentace - volba formátu. Popis instalace.

Aplikaci jsem se snažil udělat co nejvíce přehlednou a intuitivní, aby co nejvíce uživatelů aplikace pokud možno vůbec nepotřebovalo příručku. Avšak v začátcích bude jistě potřeba poradit a pro tyto účely je vytvořena i tato uživatelská dokumentace. Nejprve tedy popíšeme instalaci, a poté základní kroky s programem. Součástí dokumentace pak jsou i screenshoty s vlastní aplikací, které byly pořízeny na vývojovém počítači s Mandrivou 2010.

7.1 Instalace programu

Program jako takový se nemusí nijak instalovat, pouze se musí poprvé nastavit, aby mohl korektně fungovat. Avšak program vyžaduje nainstalovaný a správně nastavený MySQL server obsahující databázi programu. Současná verze aplikace spoléhá na existenci databáze - neumí ji sama vytvořit. Instalace MySQL serveru zde ukázána nebude, instalace je totiž různá dle pltaformy, kterou používáte, pro přesné instrukce tak vyhledejte dokumentaci ze stránek výrobce MySQL. Pro korektní chod je však nutné mít nainstalovanou Javu 6, minimálně v podobě JRE balíčku.

Jakmile máte MySQL server nainstalovaný a nakonfigurovaný, pak již jen stačí do tohoto MySQL serveru nahrát potřebnou základní či rozšířenou databázi. Základní databáze obsahuje pouze prázdné tabulky, rozšířená databáze pak obsahuje dva ukázkové filmy a několik ukázkových uživatelů. Oba typy databází najdete na CD, které je přílohou této práce.Import databáze je velmi jednoduchý, stačí se pomocí MySQL administrátora přihlásit jako uživatel root a pomocí nabídky „Restore“ v levém menu načíst Vámi vybraný typ databáze (základní či rozšířený).

Jakmile jste správně naimportovali předpřipravenou databázi, můžete začít používat program, **při prvním startu se však musí nadefinovat přístup k MySQL databázi** (tlačítko „Nastavení databáze“, které najdete na přihlašovací obrazovce aplikace), a také vytvořit uživatele (pokud jste naimportovali rozšířenou databázi, v programu již extistují dva uživatelé, uživatel „a“ a uživatel „sen063“, oba mají heslo „a“ a uživatel „a“ je zároveň administrátorem). Vše je vysvětleno v následující kapitole, viz str. 52.

Alternativní cestou pak je připojení k mé školní databázi. Jako přihlašovací údaje použijte (vše samozřejmě bez uvozovek):

- Adresa MySQL serveru: „databaze.cs.vsb.cz“
- Jméno databáze: „sen063“
- Jméno uživatele: „sen063“
- Heslo: „360nes“ (neboli můj login pozpátku)

V tomto případě však doporučuji aplikaci zkoušet přímo ve školní síti (např. na pracovním počítači ve škole), při připojení z domu je potřebné připojení přes školní VPN a

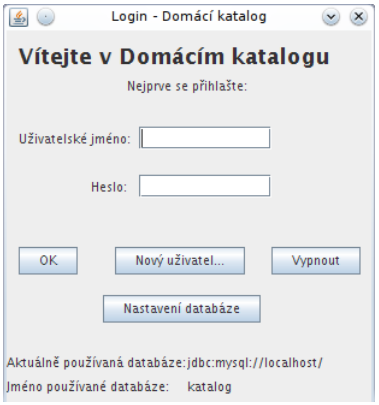
vlivem tohoto je aplikace pomalá. Tato databáze bude smazána po ukončení mého studia. I v tomto případě je samozřejmě nutné po prvním spuštění aplikace nadefinovat připojení k datbázi.

7.2 První spuštění aplikace

Jak již bylo zmíněno v předchozí části, při prvním spuštění aplikace je potřeba provést minimálně jeden krok pro úspěšných chod aplikace - nastavení připojení k MySQL databázi. Druhý krok, vytvoření uživatele (uživatelů) je již pouze doporučeným krokem a to jen v případě, kdy jste v předchozí části naimportovali rozšířenou databázi, v opačném případě je tento krok také povinným, protože by jste se jinak nemohli pochopitelně přihlásit do programu.

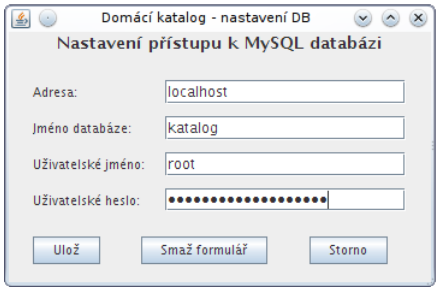
V předchozí části je také zmíněno, že aplikace nevyžaduje instalaci, je jen nutné rozbalit zip archív s aplikací (najdete jej v adresáři aplikace na CD přiloženém k této práci) kamkoli na Váš pevný disk a mít nainstalovanou podporu pro Javu. Nyní stačí klasickým dvojtým poklepáním spustit aplikaci.

Pokud bylo vše provedeno správně, měla by se Vám ihned zobrazit přihlašovací obrazovka aplikace, kterou si můžete prohlédnout i níže.



Obrázek 18: Přihlašovací obrazovka aplikace

Pro nastavení připojení k MySQL databázi použijte tlačítko „Nastavení databáze“, měla by se Vám zobrazit nová obrazovka, pomocí které nastavíte připojení k databázi. Do předpřipravených kolonek tak stačí napsat pouze adresu serveru, jméno databáze (katalog), uživatelské jméno a heslo (obojí samozřejmě k Vašemu MySQL serveru). Vyplněný formulář si můžete prohlédnout níže.



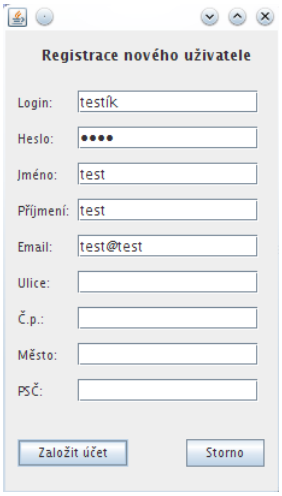
Obrázek 19: Konfigurace připojení k MySQL databázi

Nastavení se ukládá do textového souboru do adresáře, který se vytvoří, pokud neexistuje. Pokud jsou tedy nastavení jednou uložena, budou vždy použita a není je tedy potřeba modifikovat. Modifikace je samozřejmě taky dostupná, hodí se v případě, kdy změníte MySQL server.

Po úspěšném nastavení přístupu k MySQL serveru tak je potřeba již jen vytvořit uživatele. První vytvořený uživatel je vždy administrátor, další uživatelé jsou již jen obyčejní uživatelé. Samozřejmě není problém obyčejné uživatele povýšit na administrátory a naopak, administrátory nastavit jako obyčejné uživatelé, musí však být v programu vždy

alespoň jeden administrátor. Nového uživatele vytvoříte kliknutím na tlačítko „Nový uživatel“.

Následně by se Vám měl zobrazit registrační formulář, který stačí vyplnit a uživatel je ihned vytvořen. V loginu mohou být i české znaky (háčky, čárky), adresa uživatele je nepovinný údaj. Příklad vyplněného registračního formuláře si můžete opět prohlédnout níže.



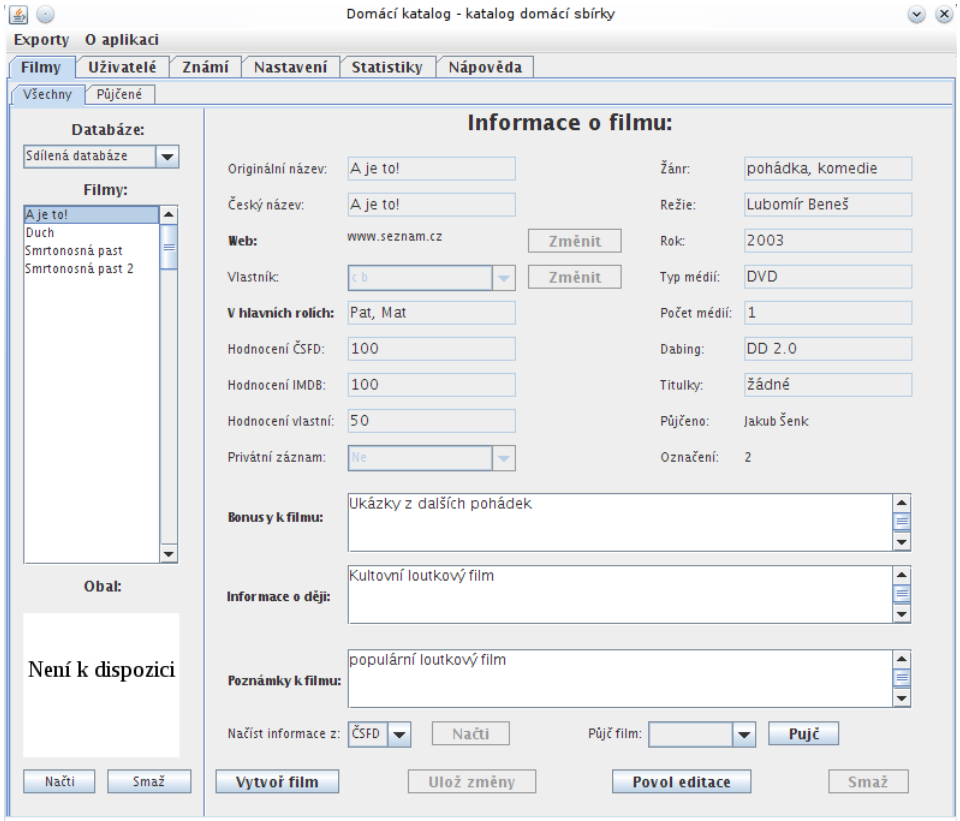
Obrázek 20: Registrace nového uživatele

Po stisknutí tlačítka „Založit účet“ je ihned uživatel vytvořen a vy se tak můžete přihlásit do aplikace. Toto provedete vepsáním přihlašovacích údajů do přihlašovacího formuláře.

7.3 Průvodce aplikací

Na několika následujících stránkách bude popsáno prostředí programu. Z kapacitních důvodů zde však nebudou uvedeny screenshoty ke všem obrazovkám programu - práce by se neuměrně nafoukla (jen základních screenshotů by tak bylo 15). Výchozím stavem pro následující řádky je, že jste nainstalovali rozšířenou databázi.

Ihned po přihlášení Vás uvidítá hlavní okno programu, ve kterém budete zároveň trávit i nejvíce času. Jako první zde uvidíte seznam již zaevidovaných filmů.



Obrázek 21: Hlavní formulář aplikace

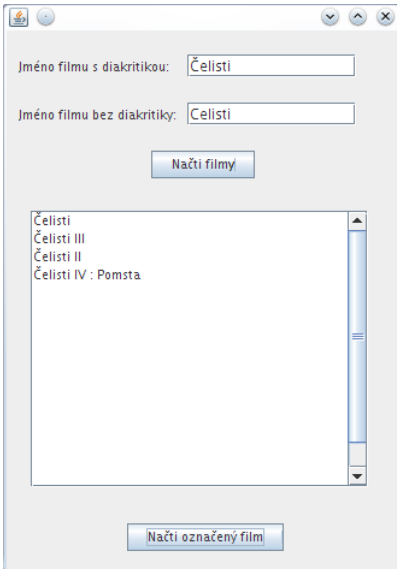
Můžete si tak zde prohlédnout v levém sloupečku seznam evidovaných filmů, pod tímto seznamem pak vidíte obal DVD. V pravé části okna pak vidíte detailní informace o vybraném filmu z levého sloupce. Pod těmito informacemi se pak nachází několik tlačítek, které slouží k manipulaci s filmy. Pokud kliknete na jmenovku pole, která je tučným fontem zvýrazněná, obsah daného pole se Vám zobrazí ve speciálním okně, čímž jsem se snažil vyřešit problém, když např. seznam herců se nevejde celý do viditelné části komponenty.

Pomocí roletkového menu „Databáze“ se lze přepínat mezi sdílenou a privátní databází. Při každé výběru v tomto roletkovém menu se uživateli nahraje daný seznam filmů. Měnit privátní filmy nemůže nikdo, ani administrátor, vracet je může buď konkrétní uživatel, nebo jakýkoli administrátor.

7.3.1 Vložení filmu

Prvním z nich je tlačítko „Vytvoř film“, pomocí kterého samozřejmě budete moci vložit do databáze nový film. Při prvním kliknutí se obsah formuláře vymaže a uživatel tak bude moci vepsat potřebné údaje. Některé zadávané údaje (seznam herců, režisérů, děj filmu, hodnoceni csfd, www odkaz) si může nechat načíst ze serveru csfd.cz, tohoto lze docílit kliknutím na tlačítko Načti, které je dostupné při vytváření či editaci filmu.

Pokud se tedy uživatel rozhodne využít této funkce, zobrazí se mu nové okno, do kterého stačí vepsat počáteční české jméno filmu s diakritikou, a poté bez diakritiky. Poté stačí kliknout na tlačítko „Načti filmy“, čímž se do seznamu načtou všechny nalezené filmy. Uživatel si vybere hledaný film a informace o tomto filmu si nechá načíst do formuláře kliknutím na tlačítko „Načti označený film“. Načtené informace se automaticky zobrazí do vyplňovaného formuláře o přidávaném filmu.



Obrázek 22: Hlavní formulář aplikace

Jakmile je uživatel spokojen, tvorbu nového filmu ukončí opětovným kliknutím na tlačítko “Vytvoř film“, čímž je film uložen do databáze. Po vytvoření filmu může daný film opatřit i obalem.

7.3.2 Editace filmu

Editace stávajícího filmu probíhá na podobném principu, jako jeho tvorba. Uživatel nejlprve vybere daný film, který chce editovat. Poté klikne na tlačítko povol editace, čímž se opět aktivují prvky formuláře a uživatel tak může provést požadované změny (id_filmu nelze změnit). Jakmile je uživatel spokojen, změny uloží tlačítkem „Ulož změny“. Změna

vlastníka, webové stránky a obalu DVD se provádějí zvlášť a pouze tehdy, když jsou povoleny změny.

7.3.3 Smazání filmu

Vybraný film lze i smazat, toto se provede označením daného filmu, který chceme smazat, v levém menu, a poté kliknutím na tlačítko „Smaž film“.

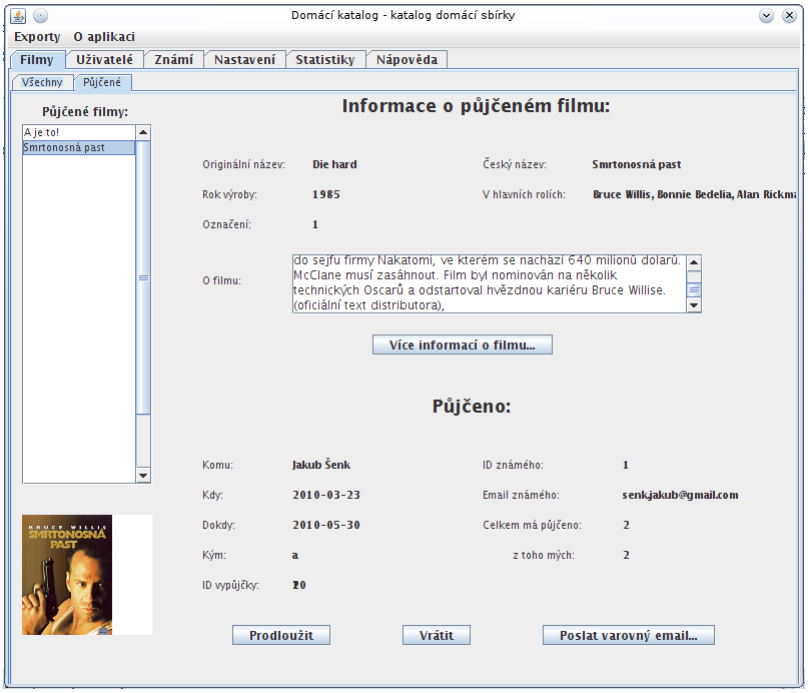
7.3.4 Půjčení filmu

V tomto okně se provádí i půjčování filmů, stačí z příslušného roletkového menu vybrat známého, kterému chceme aktuální film půjčit a kliknout na tlačítko „Půjčit“, čímž provedeme v systému vlastní půjčení. Film, který je již půjčený nelze opět půjčit.

7.3.5 Správa vypůjček

Všechny činnosti s vypůjčkami, kromě vlastního vypůjčení, se prováději v podzáložce „Půjčené“. Tato pozáložka je podobná podobná předešlé. Opět se v levé části nachází seznam, tentokrát se v něm nachází všechny aktivní vypůjčky, čili ty, které ještě nebyly vráceny. V pravé části se pak uživatel dozví nejprve něco o tom konkrétním filmu, který je půjčen a níže se nachází i informace o vlastní vypůjčce (odkdy je vedena, dokdy má být vrácen film, kým byla provedena, komu byl film půjčen a podobně). Opět se ve spod pravé části tohoto okna nacházejí tlačítka, pomocí kterých lze vypůjčku vrátit, prodloužit, či známému poslat varovný email.

Standardní doba půjčování je nastavena na 30 dní. Lze ji však přímo přes prostředí programu dle chuti měnit. Kliknutím na tlačítko „Více informací o filmu...“ se uživatel „přepne“ do rozhraní se všemi filmy a zobrazí se mu více informací o půjčeném filmu.



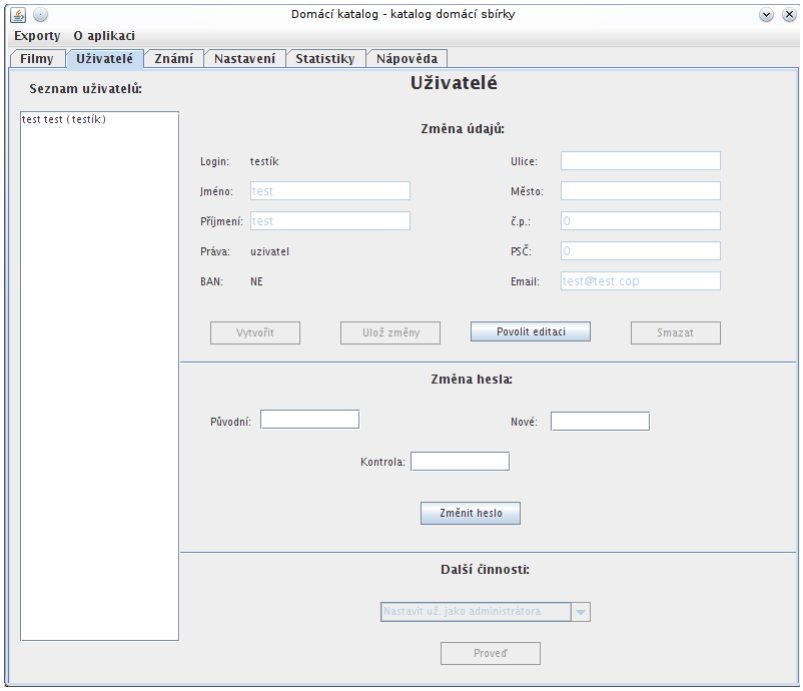
Obrázek 23: Správa vypůjček

7.3.6 Správa uživatelů

Další důležitou záložkou je správa uživatelů. V této záložce mohou administrátoři vytvářet, editovat a mazat ostatní uživatele. Obyčejní uživatelé si zde pak mohou měnit informace pouze o sobě, vymazat svůj profil a změnit si heslo.

Administrátor zde může provádět i další pokročilé úkony:

- povyšovat ostatní uživatele na administrátory
- degradovat ostatní administrátory na uživatele
- přenést všechny privátní filmy uživatele na sdílené
- mazat všechny privátní filmy uživatele
- mazat všechny filmy uživatele
- zablokovat určitý účet
- odblokovat určitý účet
- mazat účty

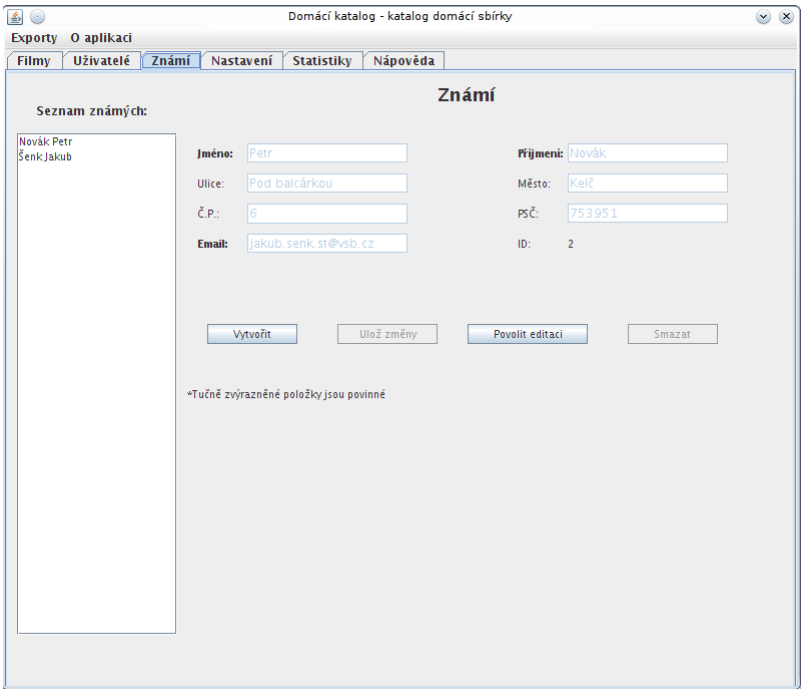


Obrázek 24: Správa uzivatelu

Tyto možnosti se nacházejí v dolním roletkovém menu. Administrátor vybere nejprve určitého uživatele, poté vybere požadovanou akci z roletkového menu a nakonec klikne na tlačítko proveď, čímž určitou akci provede. Obyčejní uživatelé sice toto roletkové okno vidí, ale je u nich neaktivní. Ke změně hesla je potřeba znát původní heslo, administrátor může měnit heslo i bez znalosti hesla původního.

7.3.7 Správa známých

Známí jsou lidé, kterým můžou uživatelé programu půjčovat filmy. Vzhled okna, pro jejich správu je opět podobný předcházejícím. Opět se v levém menu nachází seznam všech známých a v pravé části okna se zobrazují podrobnosti o vybraném známém. V tomto případě není rozlišováno mezi administrátorem a obyčejným uživatelem, oba mohou totéž - přidávat známé, editovat o nich informace a mazat je. I princip vytváření, editování a mazání uživatelů je naprosto stejný, jako v případě filmů, netřeba jej tedy popisovat.



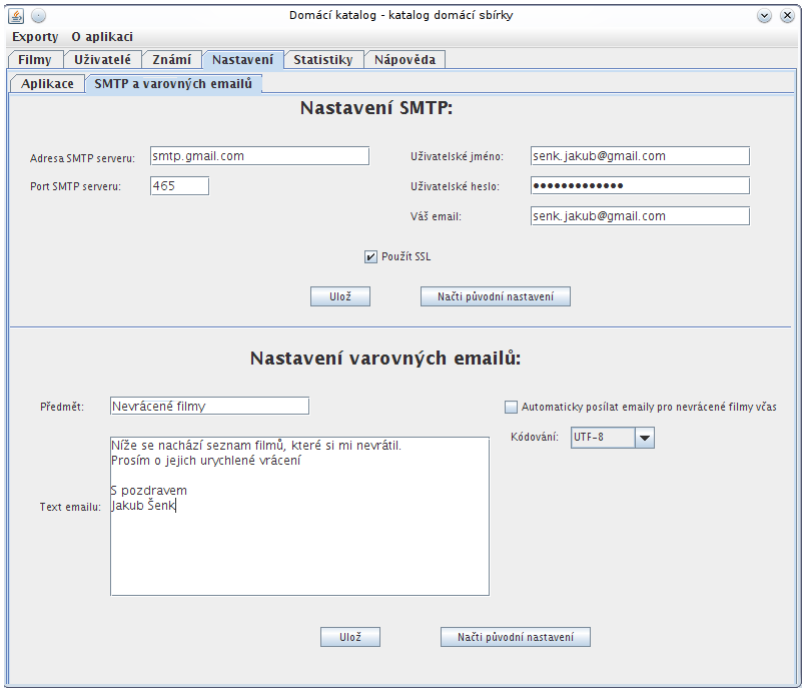
Obrázek 25: Správa známých

7.3.8 Správa emailů

Předposlední záložku, která bude popdrobněji probrána je záložka Nastavení -> SMTP a varovných emailů - neboli správu emailů. Každý uživatel totiž může posílat varovné emaily těm známým, kteří nevrátili včas nějaký film. Každý uživatel tak musí do programu zadat přihlašovací údaje ke svému emailovému účtu, skrz který chce posílat varovné emaily.

Program potřebuje znát:

- adresu SMTP serveru
- port SMTP serveru
- uživatelské jméno pro přihlášení na SMTP server
- heslo k SMTP serveru
- email
- zda-li má používat SSL
- předmět, text a kódování odesílaného emailu

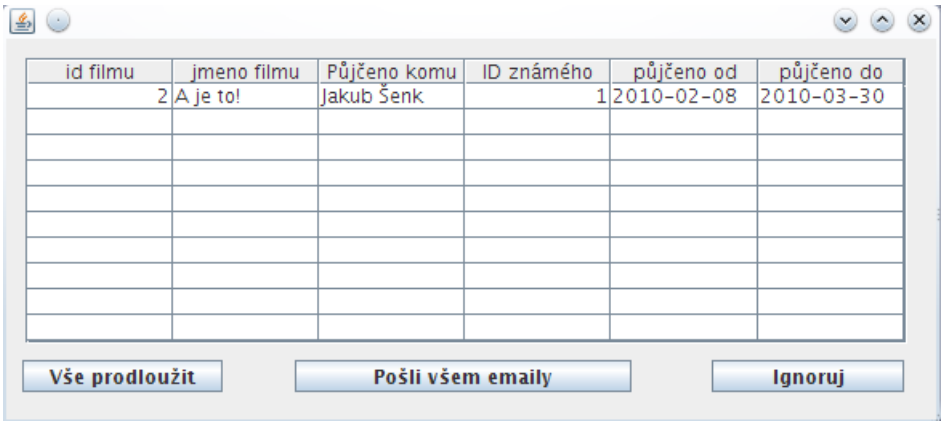


Obrázek 26: Správa emailů

Stačí tedy vyplnit předpřipravený formulář a potvrdit nastavení. Pokud jste vše zadali správně, emaily se budou posílat přes Váš emailový účet.

7.3.9 Upozorňování na včas nevrácené filmy

Poslední funkci, kterou si představíme detailněji je upozorňování na všas nevrácené filmy po přihlášení do programu. Pokud je zjištěno, že někdo nevrátil včas určitý film či filmy, program na toto ihned po přihlášení upozorní. Pokud se přihlásí obyčejný uživatel, program ho upozorní jen na nevrácené filmy, které jsou jeho privátní či jsou sdílené. Administrátora upozorní na všechny nevrácené filmy. Při přihlášení se v tom případě objeví toto upozornění:



Obrázek 27: Upozornění na včas nevrácené filmy

Uživatel tak může všechny vypůjčky prodloužit, nebo všem hříšníkům poslat varovný email nebo toto hlášení ignorovat a vyřešit přestupky přímo v programu.

7.3.10 Ostatní funkce programu

V uživatelské příručce nebyly popsány prakticky pouze dvě záložky a jedna funkce - nastavení programu, nápověda a vyhledávání. V záložce nastavení programu se nastavuje pouze vypůjční doba a cesta k internetovému prohlížeči, který zobrazí stránku libovolného filmu, který se nachází v databázi a je u něj nastavena webová stránka. Vyhledávání pak uživatel najde v nabídce „soubor“, případně se mu zobrazí po použití klávesové zkratky CTRL-ALT-F. Vyhledávat filmy může dle jeho jména (aplikace nerozlišuje mezi originálním jménem filmu a jeho českým ekvivalentem), nebo dle jména herečky či herce a také dle jména režiséra.

8 Testování, spolehlivost, možnosti rozšiřování.

8.1 Testování a spolehlivost

Testování aplikace probíhalo v první vlně na vývojovém počítači s Mandrivou 2010, který má tuto HW konfiguraci:

AMD AthlonXP 2000+ 1.5 GB RAM

Na tomto počítači byla výsledná aplikace testována po každé větší provedé změně v kódu.

V druhé vlně testování byl použit stejný počítač, akorát byla aplikace odzkoušena ve Windows XP Professional (školní licence). Tyto kontroly byli řidšího charakteru, probíhali typicky jednou za měsíc, a také vždy, když jsem šel program konzultovat s vedoucím mé Bakalářské práce.

Nejméně často bylo aplikace kontrolována přímo ve škole na školních počítačích, a to jak na linuxových kioscích (s Ubuntu 8.04) v Nové knihovně, tak i na počítačích v učebnách s Windows XP Professional. Ve všech případech se chovali stejně.

Narazil jsem pouze na čtyři problémy:

- vlivem různé geometrie grafického rozhraní napříč platformami se může stát, že ten či onen ovládací prvek může být na jedné platformě na jiné pozici než na platformě druhé, rozdíly nejsou velké, ale někdy se může stát, že malá část ovládacího prvku nejde vidět. Tomuto jsem se snažil zamezit a pevně doufám, že se tento nešvar nebude v mém programu objevovat.

- existuje i tzv. otevřená JAVA, což je JAVA udržovaná komunitou. S touto Javou osobně nemám dobré zkušenosti a vyvaroval bych se jejího využívání na počítačích s mou aplikací. S otevřenou Javou se totiž stává, že se v GUI aplikace objevují grafické artefakty.

- pokud jsem používal školní MySQL z domácího počítače, aplikace byla poměrně pomalá, což bylo způsobeno pravděpodobně školním VPN. Pokud jsem aplikaci zkoušel na jakémkoli počítači uvnitř školní sítě, a to i na notebooku spolužáka, vše fungovalo dobře a rychle.

- nepodařilo se mi najít důvod, proč se obrázky někdy nenačtou celé, ale jen z části. Proto se také po přihlášení do programu načte pouze část obalu. V seznamu půjčených DVD je také nutné vždy dvakrát kliknout na jméno filmu v seznamu půjčených DVD, jinak se opět obal nenačte celý. V ostatních případech se načte obal filmu vždy celý.

Aplikace nebyla testována na žádném volně dostupném MySQL serveru z Internetu (nenašel jsem žádný zdarma dostupný, všechny byly placené). Vždy pouze na lokálně nainstalovaném serveru, či na serveru školním.

Aplikace byla jinak stabilní a na všech platformách se chovala stejně. Podmínkou samozřejmě bylo správné nastavení aplikace, viz uživatelská dokumentace, str.51.

8.2 Možnosti rozšíření

Vlivem použitých technologií je aplikace velmi rozšířitelná. Následující text je rozdělen do dvou částí. První je zaměřena na rozšiřování aplikace jako takové (čili desktopové aplikace), druhá pak je zaměřena na další možnosti rozšiřování (webové rozhraní a podobně).

8.2.1 Rozšíření aplikace

Původně bylo plánováno širší záběr aplikace. Vlivem časové náročnosti implementace, tvorbě projektů do dalších předmětů a přípravy do výuky však byly některé funkce aplikace vyřazeny s tím, že budou naimplementovány při tvorbě případné diplomové práce či po studiu ve volném čase. Další funkce mě napadali i během vlastní implementace, některé z těchto nápadů byly přidány do stávající aplikace, případně se také přesunuly do verze budoucí.

S čím se tedy počítá v dalších verzích programu:

- hudební a softwarové databáze
- zálohování
- více typů databází, třeba pro aplikaci **pouze** lokálního a jednouživatelského charakteru nasazení databázového systému SQlite
- implementace tisku

Některé funkce by pak zasloužili vylepšit:

- exporty
- načítání přebalů - v určitých situacích se nemusí načíst celé

8.2.2 Další možnosti rozšíření

Java je velmi rozšířený programovací jazyk. Může běžet nejen na klasických počítačích, ale i na serverech a problém není ani s mobilními zařízeními. Na tomto základě se tak dá popstavit hned několik různých verzí mé aplikace a tím vlastně udělat jakousi platformu. Za použití Java EE (Java Enterprise edition) lze tak vytvořit webové rozhraní aplikace postavené na JSP stránkách a používající třívrstvou architekturu.

Typický příkladem využití může být typická situace - uživateli někdo nevrátil půjčené DVD, systém jej na to upozorní a uživatel se rozhodne, že kamaráda raději navštíví. Ten mu DVD vrátí a on se jen u něj přihlásí na webové rozhraní a vrácení provede i v systému. Změnu samozřejmě uvidí i u sebe v dekstopové aplikaci (obě verze budou používat stejnou databázi samozřejmě).

Webové rozhraní bude určeno i pro začátečníky, kterým se nechce instalovat MySQL server, tato povinnost tak je jednou z mála nevýhod použitých technologií současné aplikace. Začátečníkům tak bude stačit pouhá registrace na webových stránkách.

Jak již jsem napsal v úvodu této části, Java běží na více typech zařítení, díky Java ME (micro edition) tak umožňuje běh i na mobilních zařízeních. V daleké budoucnosti by tak mohlo být uvažováno i o vytvoření aplikace pro mobilní zařízení. Tento segment trhu je aktuálně na vzestupu a osobně si dovedu představit využití takové aplikace.

Výše uvedené by pak pomohlo k vysoké mobilitě programu. Jednalo by se tak pravděpodobně o ojedinělý produkt a případné uvolnění mezi další uživatele by ukázalo, zda-li se takový produkt uchytí.

9 Závěr

Mým cílem bylo vytvořit multiplatformní a přenositelný program pro správu domácí sbírky filmů na různých médiích, především pak na CD a DVD. Po celou dobu vývoje jsem se snažil, aby byl program co nejpřehlednější a nejefektivnější. Výsledkem mé práce je, pevně doufám, stabilní a přehledný program, který splňuje prakticky vše, co jsem si v úvodu práce předsevzal. Díky použitým technologiím není uživatel vázán na konkrétní operační systém a může použít svůj oblíbený, jedinou podmínkou je, aby na tomto operačním systému běžela Java a MySQL.

Další výhodou je, že existuje pouze jedna verze pro všechny platformy, uživateli tak vždy stačí jeden balíček s aplikací. Mezi další výhody nesporně patří i fakt, že narozdíl od konkurence používá klasickou databázi typu klient-server (a ne embedded), díky které je aplikace více rozšířitelná. V budoucích verzích tak může být naimplementováno i webové a mobilní rozhraní, což by za použití embedded databáze nebylo možné. Stejně tak víceuživatelský režim je možný právě díky použití klient-server databáze.

Jednou z mála nevýhod aplikace je fakt, že uživatel musí mít nainstalovaný MySQL server. Aplikace by však mohla být v budoucnu, až by se naimplementovalo webové rozhraní, i uvedena do komerčního provozu. Uživatel by tak neplatil za program jako takový, ale za MySQL hosting. Ostatně o případné rozšířitelnosti se zmiňuji i k tomu určené kapitole, str. 65. Díky vývoji aplikace jsem si také prohloubil své znalosti z jazyka Java a také z MySQL. Naučil jsem se i používat Java knihovny, které nejsou součástí Java API, v mém případě tedy hlavně Jericho HTML parser, díky kterému jsem naimplementoval alespoň základní podporu načítání informací o filmu ze serveru http://www.csfd.cz. Během implementace jsem si tak Javu a MySQL velmi oblíbil a bude-li mi umožněno, rád bych studoval programování i v navazujícím studiu.

Jakub Šenk

10 Reference

[1] Brůha, Lubomír, *Java - Hotová řešení*, Brno: Computer press, 2003. ISBN 80-251-0072-3

[2] Molinaro, Anthony, *SQL kuchařka programátora*, Brno: Computer press, 2009, ISBN 978-80-251-2617-2.

[3] Definice multiplatformnosti na české Wikipedii
http://cs.wikipedia.org/wiki/Multiplatformnost

[4] SQLite na české Wikipedii
http://en.wikipedia.org/wiki/SQLite

[5] JAVA 6 SE API
http://java.sun.com/javase/6/docs/api/

[6] Java mail API
http://java.sun.com/products/javamail/javadocs/index.html

[7] Ukázkový příklad parsování HTML stránky pomocí Jericho HTML parseru
http://jericho.htmlparser.net/samples/console/src/ExtractText.java

[8] Jericho html parser API
http://jericho.htmlparser.net/docs/javadoc/index.html

[9] JExcel ukázkový příklad tvorby XLS souboru (3 až 5 odstavec)
http://rgagnon.com/javadetails/java-0516.html

[10] JExcel API
http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/